量子コンピュータ入門

-- 量子コンピュータと人工知能 --

Part I
量子コンピュータの基礎

- 量子ビット -- 量子の状態をベクトルで表現する
- 量子ゲート -- 量子の状態変化を行列で表現する
- 量子回路 -- 複数の量子の状態をテンソルで表現する
- 量子の不思議 1 – 量子の状態は観測できない？
- 量子の不思議 2 – エンタングルメント

Part II
量子アルゴリズムの基礎

- 量子コインと量子暗号
- Quantum Parallelism
- エンタングルメントと量子通信

Part III
量子コンピュータの歴史と到達点

- ファインマン
  1982年　自然をシミレートする量子コンピュータ
- ショア
  1994年　量子アルゴリズムの力
- ローズ (D-Wave)
  2011年　量子コンピュータ実現の困難さと新しい道
- Google（マルチネス）
  2019年　量子超越性の実証

Part IV Agenda
量子コンピュータと人工知能論

- 「機械は考えることができるか？」
- 現代の生命観 -- 分子機械としての人間
- 機械の知能の計算主義モデル
- 複雑性理論の知見
- 人間と機械の「共生」について

Part I

量子コンピュータの基礎

Part I Agenda
量子コンピュータの基礎

1. 量子ビット -- 量子の状態をベクトルで表現する
2. 量子ゲート -- 量子の状態変化を行列で表現する
3. 量子回路 -- 複数の量子の状態をテンソルで表現する
4. 量子の不思議 1 – 量子の状態は観測できない？
5. 量子の不思議 2 – エンタングルメント

参考資料
「紙と鉛筆で学ぶ量子コンピュータ入門演習」
https://www.marulabo.net/docs/q-seminar2019/

量子ビット

-- 量子の状態をベクトルで表現する --

量子ビット
-- 量子の状態をベクトルで表現する --

- bitと量子bit
- 重ね合わせ Superposition
- ケット |k>のたとえ話
- 「重ね合わせ」のたとえ話
- 「重ね合わせ」は量子だけではない
- **bitとqubit --** 古典状態と量子状態の違い

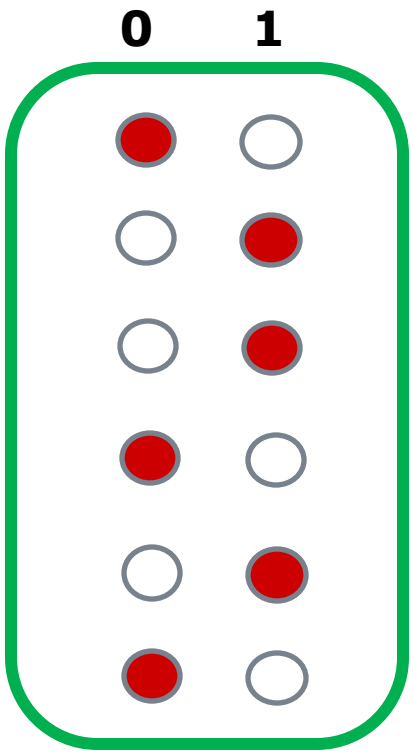
bitと量子bit

bitは、0または1の値を取る

bitの例

0
1
1
0
1
0

bitの取る値

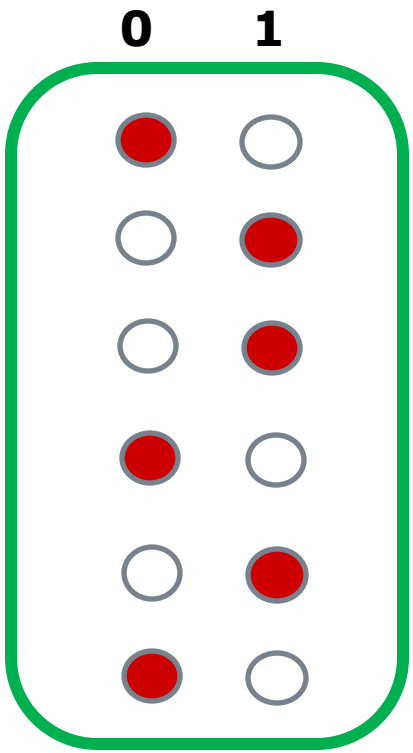


bitは、0または1の値を取る

bitの例

0
1
1
0
1
0

bitの取る値



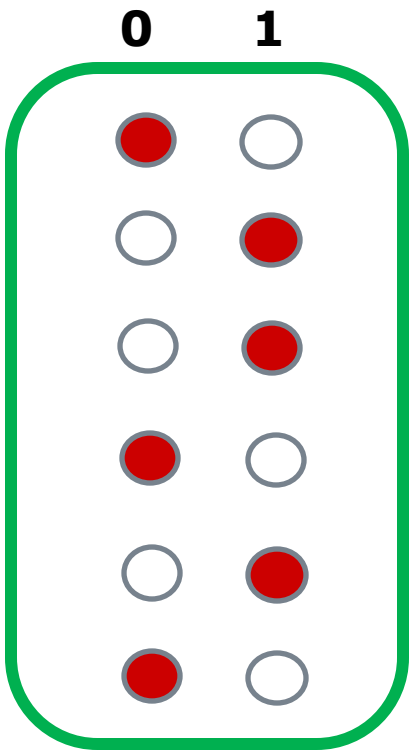
bitは、0または1以外の値を取らず、離散的である。

bitは、0または1の値を取る

bitの例

0
1
1
0
1
0

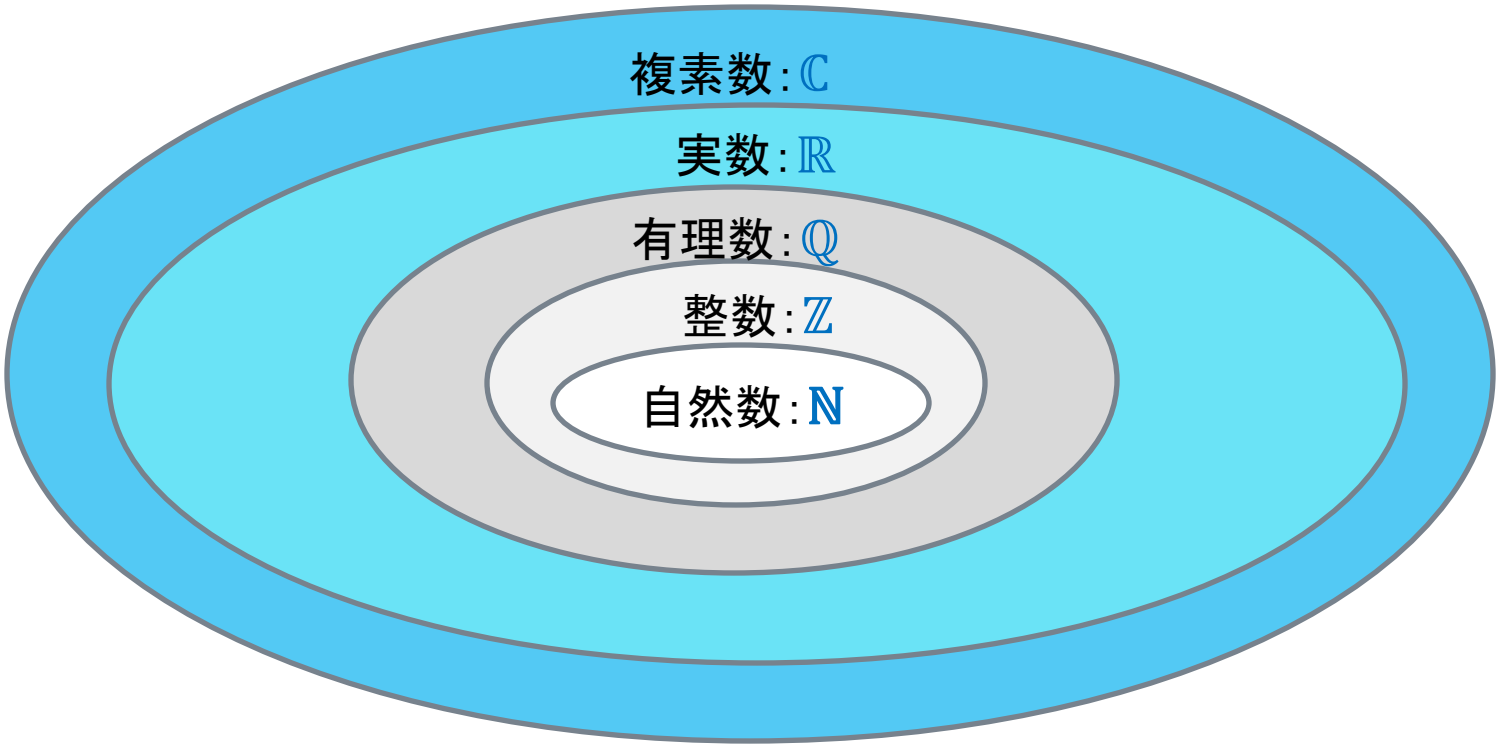
bitの取る値



bitは、0または1以外の値を取らず、離散的である。
bitは、一つの数字で表される

スカラーとベクトル

- 一つの「数字」で表される量をスカラーといい、二つ以上の「数字」の組で表される量をベクトルという。（単純化している）
- ここでの「数字」は、次のものを含む。



量子ビットは、二つの数字で表現される

量子ビット qubit は、もっとも単純な量子の状態である。それは、二つの数字の組みで表現される。量子ビットは、ベクトルとして表現される。

bitの状態 ＝ 0か1

0か1のみの値を取る、一つの数字で表される

qubitの状態 ＝ $\begin{pmatrix} a \\ b \end{pmatrix}$

aとb 2つの数字で表されるベクトル

（ただし、a,bは複素数で、$|a|^2+|b|^2=1$）

一次元の点としての古典bit
二次元のベクトルとしての量子bit

- 古典bitは、0 または 1 の値しかとらない二つの離散的な点として表現される。
- 量子bit qubitは、|0>と|1>という二つのベクトルによって張られ、aとbという二つの数によって決定される二次元のベクトルとして表現される。この状態を「重ね合わせ」という。

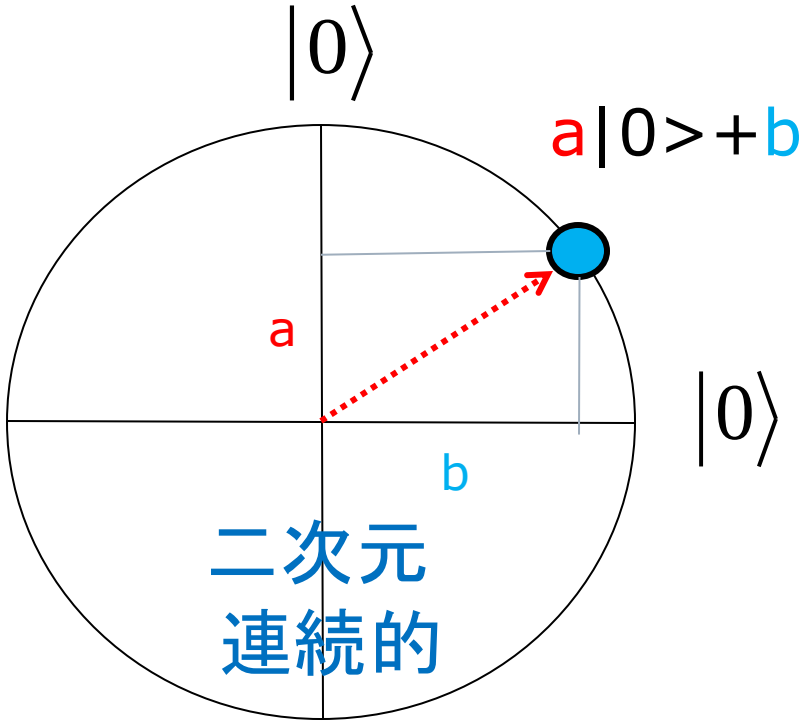
$$|Qubit> = a|0> + b|1>$$

この時、$|a|^2+|b|^2=1$ という条件がつく。
a, b は、複素数の値を取る。

- qubitは、二次元の複素ベクトルとして表現される。

二つのビットは、異なる性質を持つ

qubit



qubitは、|0>と|1>の「重ね合わせ」で、aとbの二つの数で決まる

$$\begin{pmatrix} a \\ b \end{pmatrix}$$

a：|0>の成分

b：|1>の成分

という二次元ベクトルである。

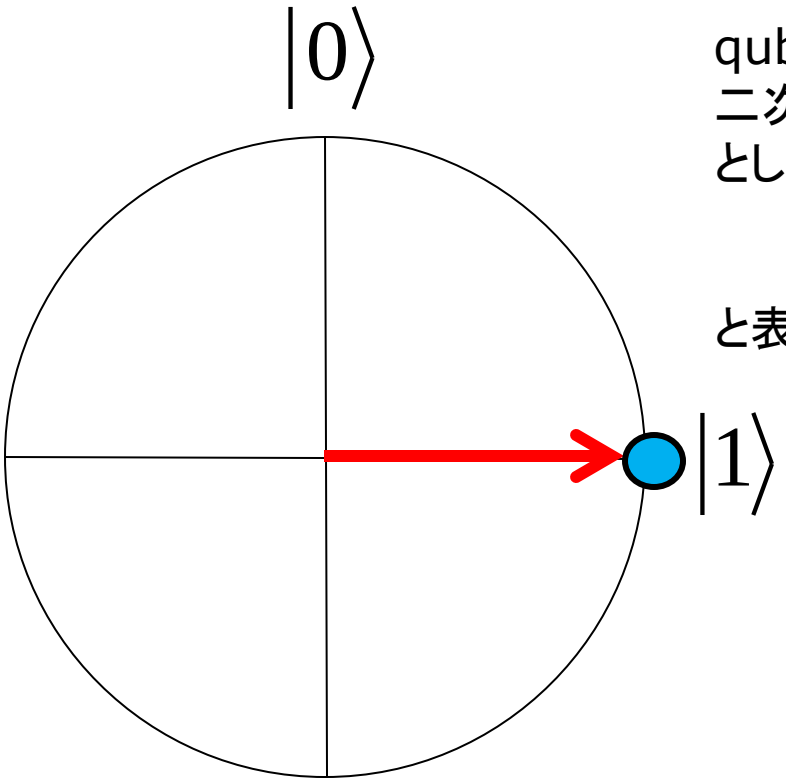
$|a|^2+|b|^2=1$

である。

bit

一次元
離散的

|Qubit> = a|0>+b|1>
で、a=0, b=1の時の状態



qubit |1>は、
二次元のベクトル
として、

$$|1> = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

と表される

qubit $|1\rangle$ と bit 1 は、対応している

qubit

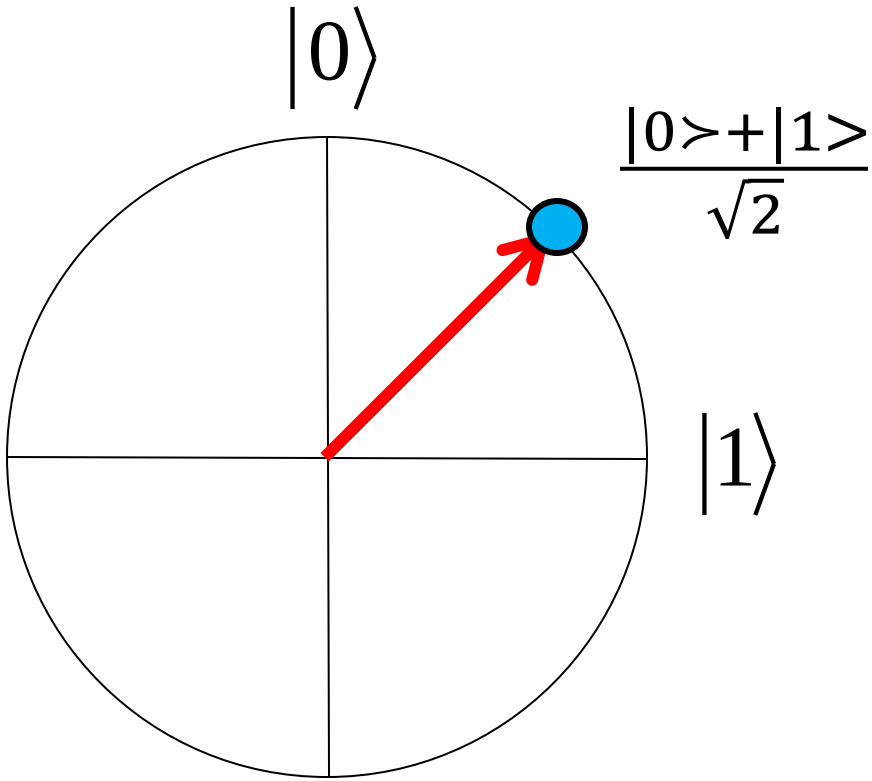
0 1

bit

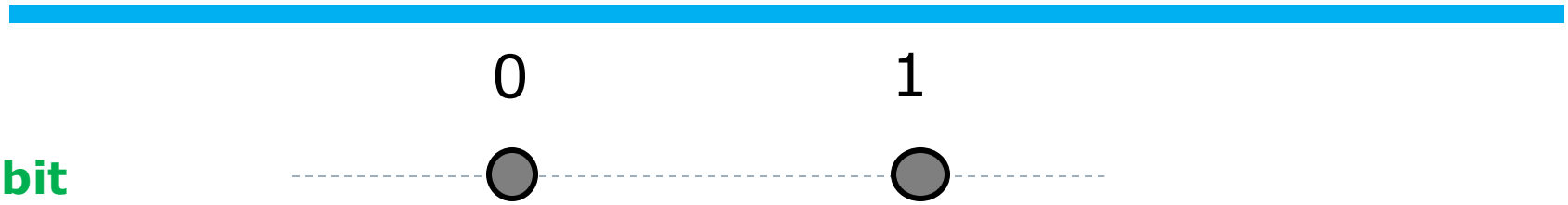
|Qubit> = a|0>+b|1>

で、a=$\frac{1}{\sqrt{2}}$, b= $\frac{1}{\sqrt{2}}$ の時の状態

qubit

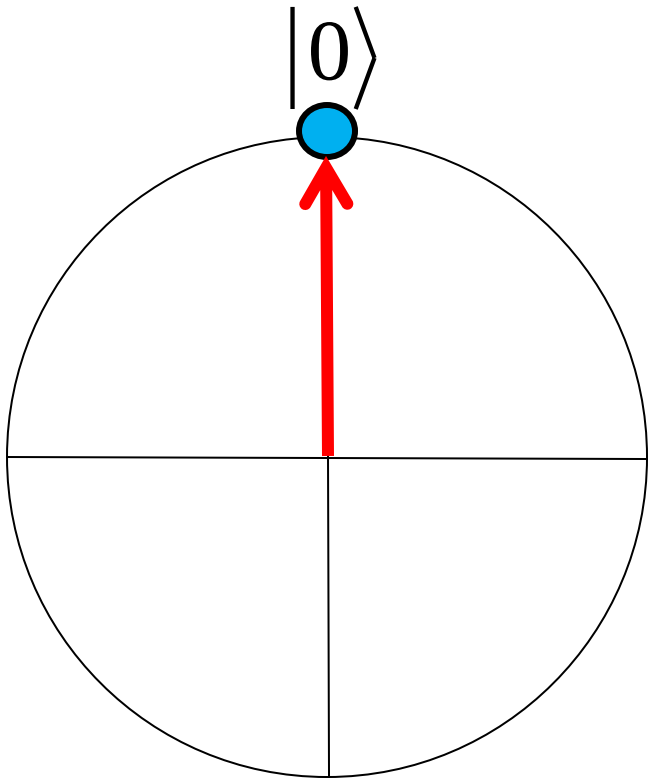


bit



qubit と bit は、対応しない

|Qubit> = a|0>+b|1>
で、a=1, b= 0の時の状態



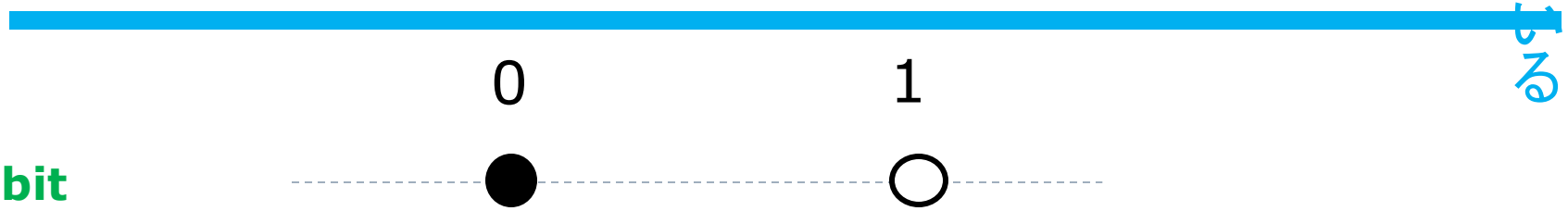
qubit |0>は、
二次元のベクトル
として、

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

と表される

qubit $|0\rangle$ と bit 0は、対応している

qubit



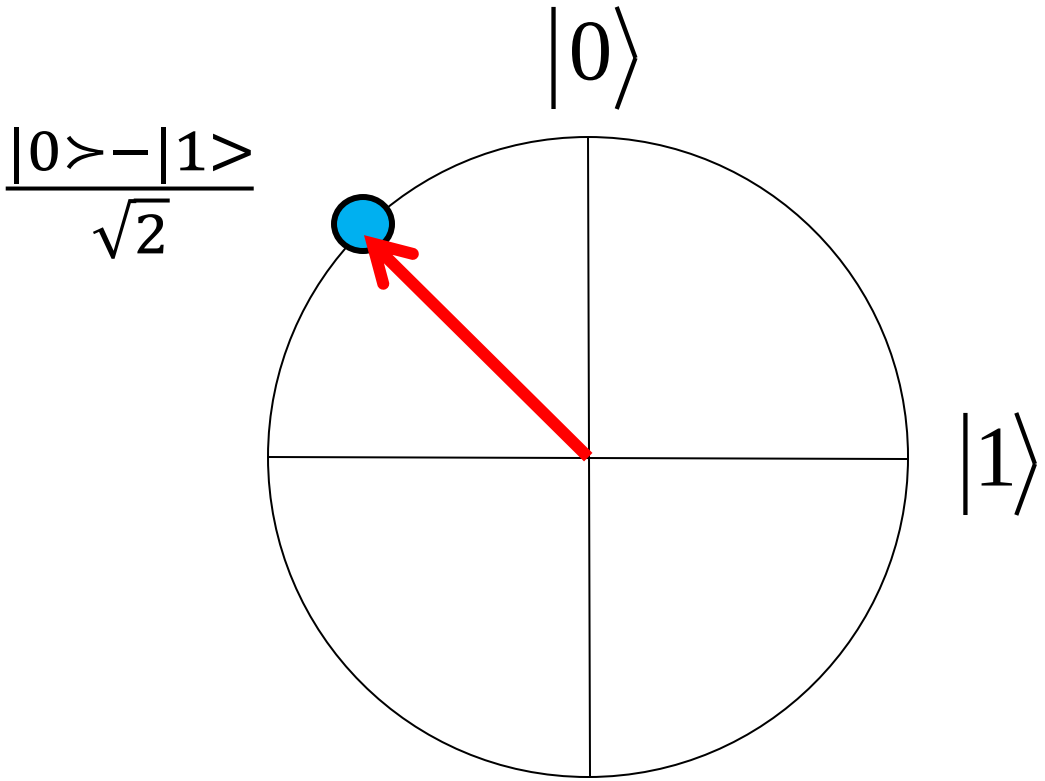
bit

|Qubit> = a|0>+b|1>

で、a=$\frac{1}{\sqrt{2}}$, b= $-\frac{1}{\sqrt{2}}$ の時の状態

qubit と bit は、対応しない

qubit



重ね合わせ Superposition

重ね合わせ
量子ビットは、二つのベクトルの和として表される

qubitの状態 |Q> $= \begin{pmatrix} a \\ b \end{pmatrix} = \underbrace{\begin{pmatrix} a \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ b \end{pmatrix}}$

これは、$\begin{pmatrix} a \\ 0 \end{pmatrix}$と$\begin{pmatrix} 0 \\ b \end{pmatrix}$という二つのベクトルの和である

このことを、二つの状態の「重ね合わせ」という

これは、さらに次のように変形できる。

$$= a\begin{pmatrix} 1 \\ 0 \end{pmatrix} + b\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

スカラー　　ベクトル

ケット記法 |0>と |1>

qubitの状態|Q>は、$a|0> + b|1>$で表せる

ここで、 ベクトル $\begin{pmatrix}1\\0\end{pmatrix}$ を |0>,

ベクトル $\begin{pmatrix}0\\1\end{pmatrix}$ を |1> で表すと

qubitの状態 $|Q> = \begin{pmatrix}a\\b\end{pmatrix} = a\begin{pmatrix}1\\0\end{pmatrix} + b\begin{pmatrix}0\\1\end{pmatrix}$

$= a|0> + b|1>$

と表すことができる

二次元のすべてのベクトルは、二つのスカラー a,b を用いて、この形で表現できる。
こうしたとき、|0>, |1> を、このベクトル空間の「基底」と呼ぶ。

（ただし、qubitの場合、*a,b*は複素数で、$|a|^2+|b|^2=1$ である）

もっと一般に、量子の状態 |ψ>は、n次元の複素数ベクトルで表現される

$$\text{量子の状態 } |\psi\rangle = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{pmatrix}$$

ただし、$c_0, c_1, \ldots, c_{n-1}$は複素数

こうした複素数ベクトル空間をヒルベルト空間という

もっと一般に、量子の状態 |ψ>は、n個の状態の重ね合わせとして表現される

n次元複素ベクトル空間の基底を |0>,|1>,|2>,…,|n-1> とすると、

量子の状態 $|\psi\rangle = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{pmatrix}$ は、

$$|\psi\rangle = c_0|0\rangle + c_1|1\rangle + \ldots + c_{n-1}|n-1\rangle$$

と表現される。ただし、$|c_0|^2+|c_1|^2+\ldots+|c_{n-1}|^2=1$

量子の状態 $|\psi>$ を、Σ を使って表す

総和の記号Σ を使うと、量子の状態を見やすく表現できる。

量子の状態　$|\psi> = c_0|0> + c_1|1> + \ldots + c_{n-1}|n-1>$

$$= \sum_{K=0}^{n-1} c_k|k>$$

ただし、 $\sum_{K=0}^{n-1} |c_k|^2 = 1$

ケット |k>のたとえ話

参考資料
「たとえ話で理解する量子の世界」
https://www.marulabo.net/docs/q-talk/

ケット|・>を状態を表す記号と解釈してみよう
日常語のレベルでの言い換え

- ここで、状態を表す記号 |・>記号を導入して、
  「〇〇の状態」を|〇〇の状態>で、
  それがとる様々な状態を|状態>で表すことにしよう。

- 例えば、
  |健康の状態> ＝ |健康>, |健康の状態> ＝ |病気>,
  |心理の状態> ＝ |喜>, |心理の状態> ＝ |怒>,
  |心理の状態> ＝ |哀>, |心理の状態> ＝ |楽>, …

- この式は、「数学的」なものではないとは言えない。それは、普通の言語で簡単に言い換えることができる。

|生活の状態> ＝ |貧しい>
形式化してわかること

- 例えば、|生活の状態> ＝ |貧しい> という式を考えてみれば分かるように、同じ記号を使っているが、左辺の|状態>と右辺の|状態>とでは、表している意味が違うということである。
  普通の言語に直してみれば、左辺は主語で右辺は述語である。

- こうした関係は、プログラミング言語や数学での「変数」で考えるともっと分かりやすい。左辺は、変数の名前を指して、右辺は変数の値を指している。

- 「状態」は、形式的には、「変数」に「値」を割り当てることで表現されるというイメージは、とても役に立つものである。

二つの値のみを取る質的な状態の認識

|対象の状態> ＝ |Aである>
|対象の状態> ＝ |Aでない>

あるいは

上の式は、次の式と同じである

|対象の状態> ＝ |A>
|対象の状態> ＝ not|A>

あるいは

上の式は、次のように変形できる

|対象の状態> ＝ **1**|**A**>
|対象の状態> ＝ **0**|**A**>

あるいは

Bit型

この形は、次に見る量的な状態の認識の形と共通するものである

「量的な状態」の表現
物理量の「単位」にケット|.>を使ってみる

- 状態を表すのは、質的な情報だけではない。
- 例えば、体重 68kg, 身長 168cm のような体重・身長といった物理的な状態は、単位の違いを無視すれば、一つの数字で表すことができる。単位を含めて表現すれば、次のようになる。

数　単位

- 僕の体重 ＝ 68 体重kg
  僕の身長 ＝ 168 身長cm

- ここでは、次のような表現を考えてみよう。
  |僕の体重> ＝ 68|体重kg>
  |僕の身長> ＝ 168|身長cm>

「重ね合わせ」のたとえ話

二つの状態の重ね合わせは、qubit だけではない

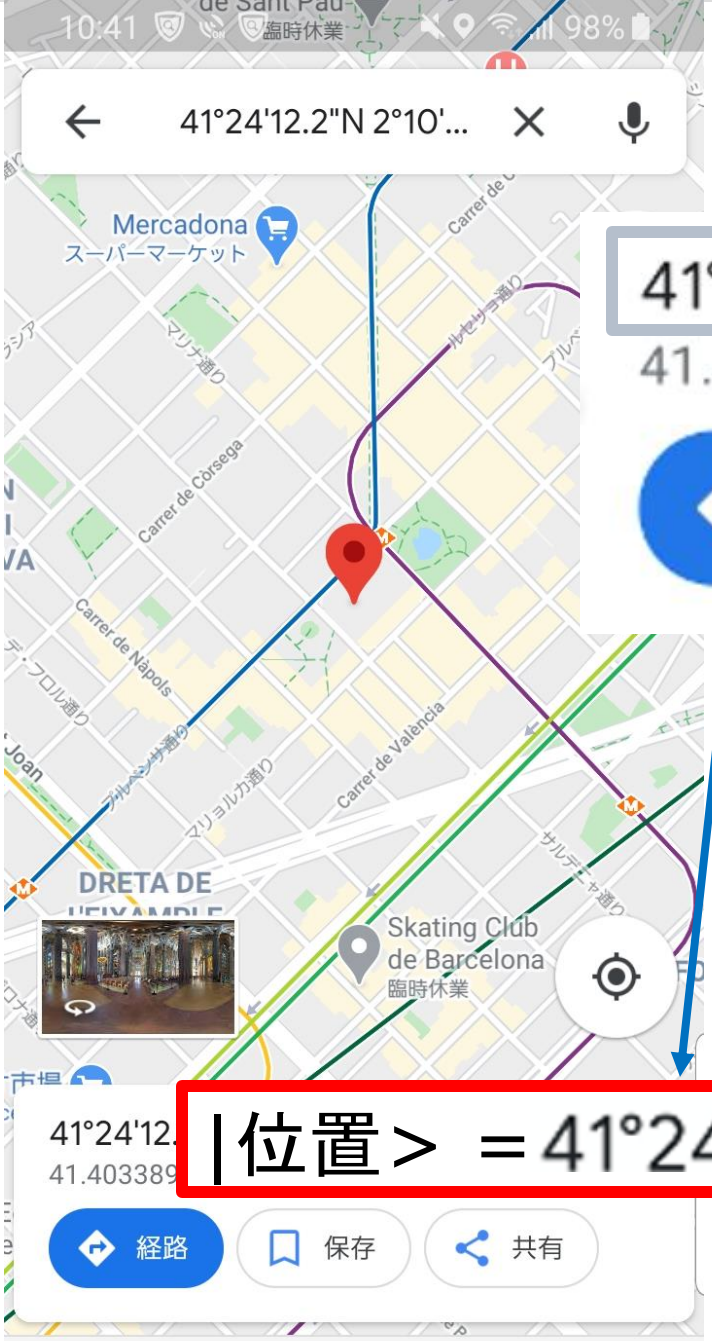
二つの数字で表現される量的状態
例1　GPSの「位置」情報

- 先にみた量的状態は、状態を表す一つの「単位」と、一つの数字で表されていた。それは、「単位」を捨象すれば、一つの数字で表されていると考えることができる。

- 先に見たように、qubitの状態を表現するには、二つの数字が必要なのだが、量子の世界でなくても、一つの状態なのに、それを表すのに二つの数字が必要な状態がある。

- たとえば、GPSで表される「位置」の情報は、北緯と東経を表す二つの数字で表されている。

二つの数字で表される「位置」



41°24'12.2"N 2°10'26.5"E

41.403389, 2.174028

経路　保存　共有

|位置> = 41°24'12.2" |北緯> + 2°10'26.5" |東経>

数値　単位　　数値　単位

5-chōme-1 Tomioka

Wakkanai, Hokkaido 097-0012, Ja...

45.389850, 141.722333

bake cafe らふ

天理教 元宗谷分教会

Wakkanai Municipal Shiomigaoka ES
稚内市立潮見が丘小

5-chōme-9-1 Tomioka

Wakkanaishi Hagigaoka Water Treatment Plant
稚内市萩ケ丘浄水場

Sun Road(サンロード)

パレット

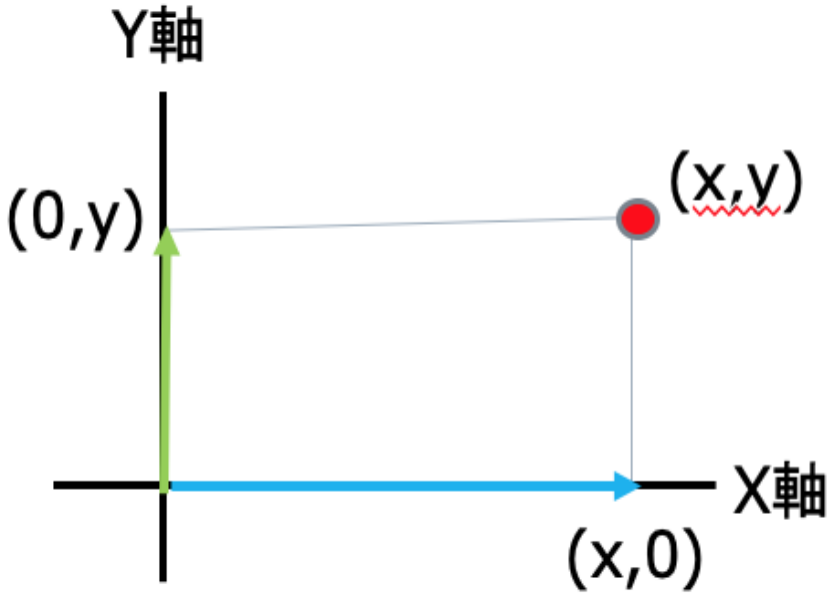
Satellite

5-chōme-1 Tomioka
Wakkanai, Hokkaido 097-0012, Ja...
45.389850, 141.722333

|僕の位置>
= 45.389850|北緯> + 141.722333|東経>

二つの数字で表現される量的状態
例2　数学的な「XY-座標」の情報

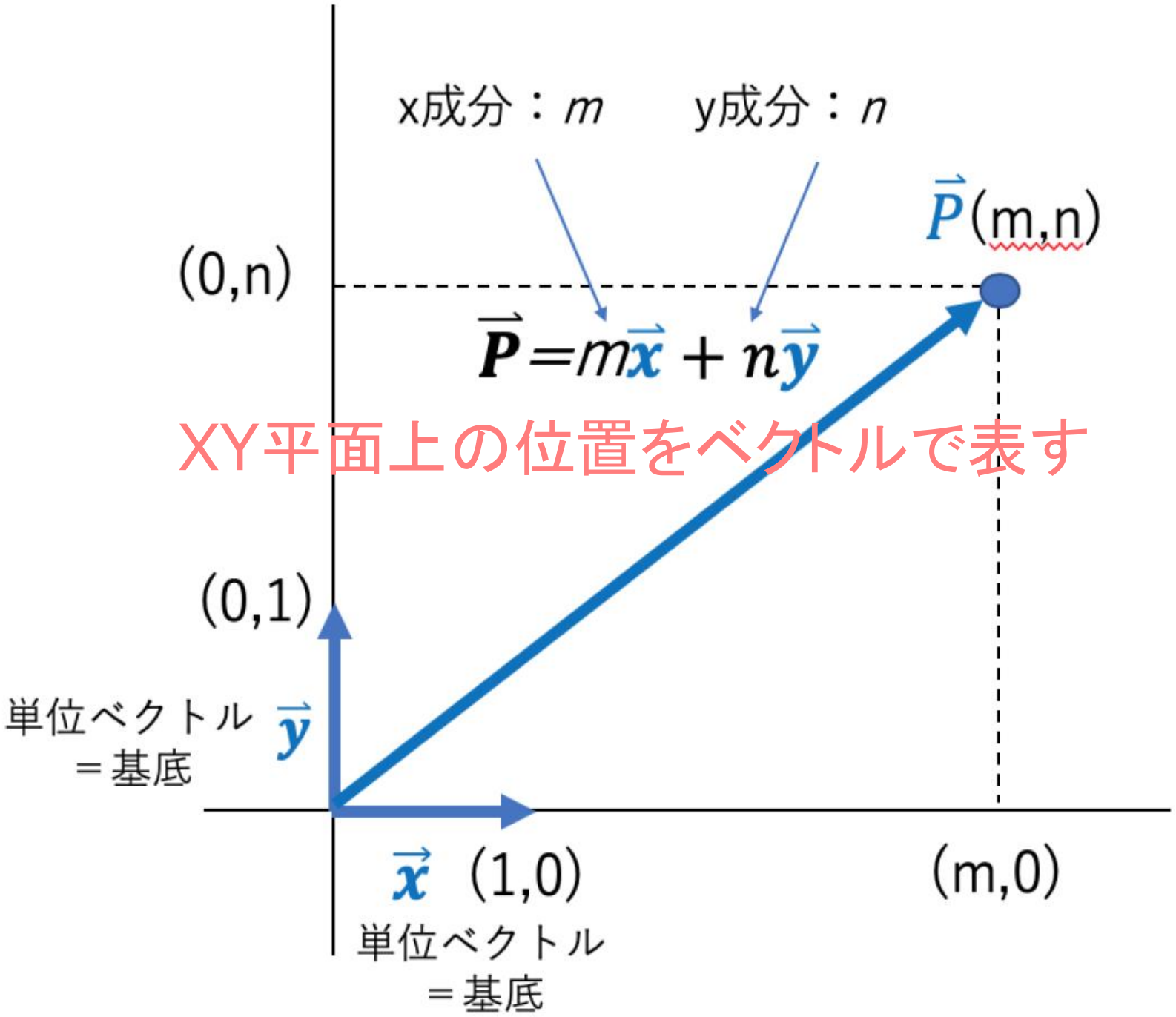
- たとえば、数学的な「XY-座標」の情報は、X座標とY座標を表す二つの数字 (x, y)で表されている。



x軸方向の成分　y軸方向の成分

$$(x,y) = (x,0) + (0,y)$$
$$= x(1,0)+y(0,1)$$

x軸方向の単位成分　y軸方向の単位成分


x成分：m
y成分：n
$\vec{P}$(m,n)
(0,n)
$\vec{P}=m\vec{x}+n\vec{y}$
XY平面上の位置をベクトルで表す
(0,1)
単位ベクトル
＝基底
$\vec{y}$
$\vec{x}$ (1,0)
単位ベクトル
＝基底
(m,0)

ベクトル $\vec{P}=m\vec{x}+n\vec{y}$ を 状態の言葉、ケット|>で表現する

- $\vec{P}=m\vec{x}+n\vec{y}$ という XY平面上の位置の表現は、次のように書き直すことができる。

$$|P> = m|x> + n|y>$$

m, n は数である。ここでも、|x>, |y>は、それぞれ「独立」なx-方向、y-方向の「単位」になっている。（単位ベクトル）

位置の状態 |P>は、二つの数で表現される

- この、式

$$|P\rangle = m|x\rangle + n|y\rangle$$

は、先に見たGPSでの次のような位置の表現と、形式的には同じものである。

$$|P\rangle = m|\text{北緯}\rangle + n|\text{東経}\rangle$$

- いずれの形式でも、位置の状態 |P>は、座標系の取り方にかかわらず、二つの数で表現されている。これは、我々が普段扱う位置の状態が持つ情報の、とても大事な特徴である。

「重ね合わせ」は量子だけではない

量子ビットの状態は、二つの数字で表される

- 量子ビットの状態|Q>は、二つの数字m. nで表される。

$$|Q\rangle = m|0\rangle + n|1\rangle$$

量子の状態は、二つの数字で表される

- 量子ビットの状態|Q>は、二つの数字m. nで表される。

$$|Q\rangle = m|0\rangle + n|1\rangle$$

これは、先に見た 次のようなXY平面上の位置の表現や、GPSでの位置の表現と同じ形をしている。

- XY平面上の位置の表現

$$|XY\rangle = m|x\rangle + n|y\rangle$$

- GPSでの位置の表現

$$|GPS\rangle = m|\text{北緯}\rangle + n|\text{東経}\rangle$$

状態の「重ね合わせ」の例

- 量子ビットの状態|Q>

$$|Q\rangle = m|0\rangle + n|1\rangle$$

- XY平面上の位置の表現

$$|XY\rangle = m|x\rangle + n|y\rangle$$

- GPSでの位置の表現

$$|GPS\rangle = m|\text{北緯}\rangle + n|\text{東経}\rangle$$

「重ね合わせ」の状態を構成するもの
数字（スカラー）「成分」

- 量子ビットの状態|Q>

$$|Q\rangle = m|0\rangle + n|1\rangle$$

成分(スカラー)

- XY平面上の位置の表現

$$|XY\rangle = m|x\rangle + n|y\rangle$$

成分(スカラー)

- GPSでの位置の表現

$$|GPS\rangle = m|\text{北緯}\rangle + n|\text{東経}\rangle$$

成分(スカラー)

「重ね合わせ」の状態を構成するもの
基本となる単位（ベクトル）「基底」

- 量子ビットの状態|Q>

$$|Q\rangle = m|0\rangle + n|1\rangle$$

基底(ベクトル)

- XY平面上の位置の表現

$$|XY\rangle = m|x\rangle + n|y\rangle$$

基底(ベクトル)

- GPSでの位置の表現

$$|GPS\rangle = m|\text{北緯}\rangle + n|\text{東経}\rangle$$

基底(ベクトル)

bitとqubit
古典状態と量子状態の違い

- 古典ビットの取る状態 0, 1 を、|0>, |1>と同一視すると、古典ビットの状態 |Bit>も、次のように表現できる。

$$|Bit\rangle = m|0\rangle + n|1\rangle$$

ただし、これではビットが 0 または 1の値しか取らないことをうまく表現できていない。それには、次のような条件を追加して、かつ、m, n は0 または 1の値しか取らないことにすればいい。

$$m + n = 1$$ (あるいは n = 1– m)

- 量子ビットの状態|Q>も、同じように表されるのだが、

$$|Q\rangle = m|0\rangle + n|1\rangle$$

成分 m, n は、次の条件に従う。

$$|m|^2 + |n|^2 = 1$$

bitは、特殊な qubit

- m は0 または 1の離散的な値しか取らないとする。
  この時、次のような状態 |B> を考える。

$$|B\rangle = \sqrt{m}|0\rangle + \sqrt{1-m}\,|1\rangle$$

$|\sqrt{m}\,|^2 + |\sqrt{1-m}\,|^2 = 1$ なので、|B>は 量子ビットである。

- m ＝ 0 の時、|B>は |1> ,
  m ＝ 1 の時、|B>は |0> と観測される。
  |B>は、古典ビットと同じように振る舞う。

- 古典ビットは、特殊な退化した量子ビットとして解釈できる。

量子ゲート

-- 量子の状態変化を行列で表現する --

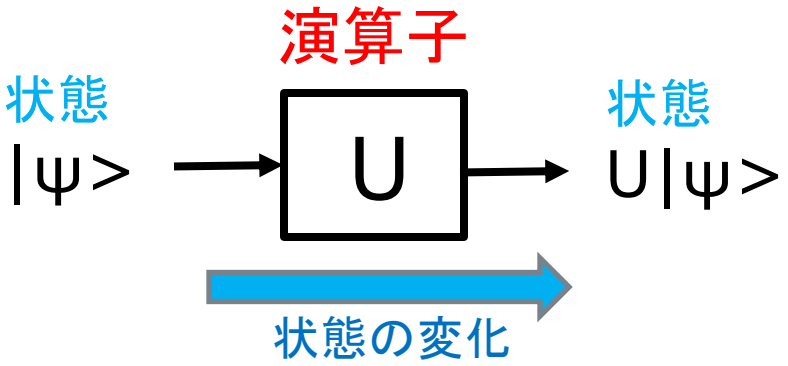
量子ゲート
-- 量子の状態変化を行列で表現する --

- 量子の状態＝ベクトル、演算子＝行列
  - 量子の状態変化を表すユニタリ行列、行列の共役
  - 行列 X, Z, Hの性質
- 量子ゲート ＝ ユニタリ行列
- 1-qubit ゲート　X, Z, H
- 2-qubitのゲート、CNOTゲートとControl-Uゲート
- 古典ゲートと量子ゲート

量子の状態＝ベクトル、演算子＝行列

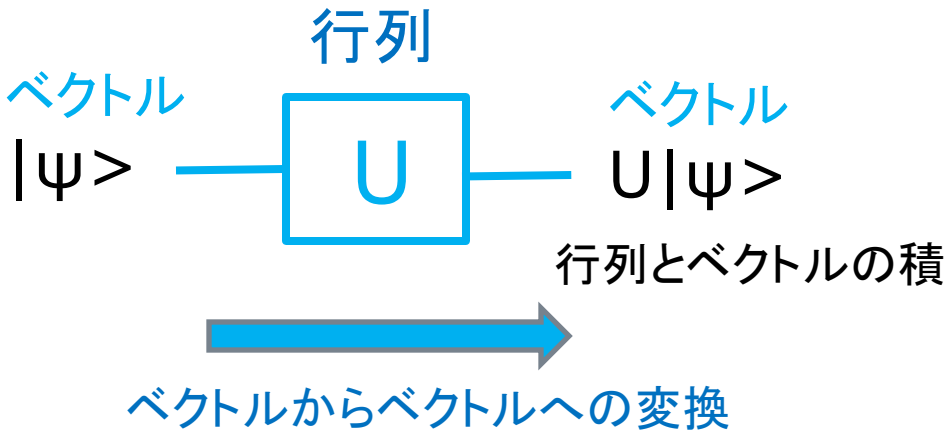
量子の状態を変化させる演算子

- 量子の状態を変化させる作用を持つものを演算子と呼ぶ。
- 量子の状態 $|\psi\rangle$は、演算子Uの作用を受けて、新しい状態 $U|\psi\rangle$ に変化すると考える。

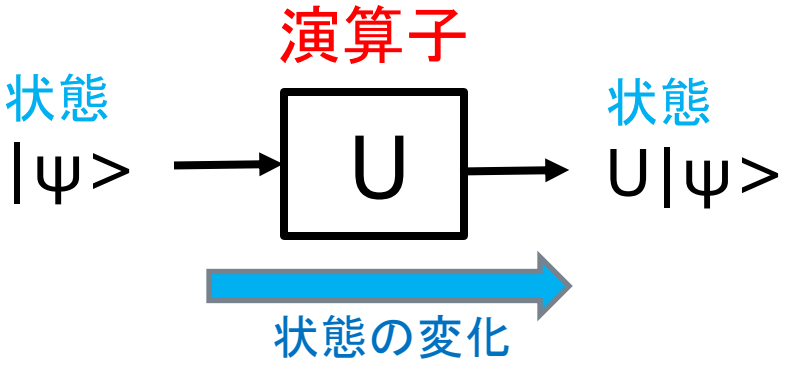


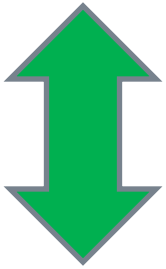
量子の状態＝ベクトル、
ベクトルからベクトルへの変換＝行列

- 量子の状態がベクトルで表されるのなら、量子の状態の変化は、あるベクトルから他のベクトルへの変化として表される。
- あるベクトルから他のベクトルへの変換は、行列とベクトルの積で表現される。ｎ ｘ ｎ の行列に n次元のベクトルを掛けるとn次元のベクトルが生まれる。

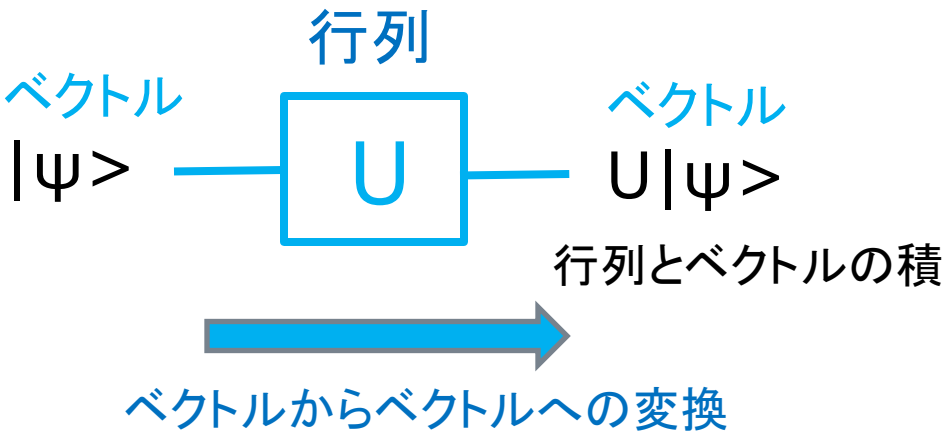


n次元のベクトルで表される量子の状態 |ψ>に作用して
状態を変化させる演算子Uは、n x n の行列とみなすことができる。





演算子 ＝ 行列



量子の状態変化を表すユニタリ行列
行列の共役

- ただし、すべての行列が、量子の状態変化を表すわけでない。量子の状態変化を表す性質を持つ行列を、ユニタリ行列という。
- ある行列Uの共役$U^{\dagger}$を、行と列を入れ替え(この操作を転置と言いTで表す)、全ての要素aの複素共役a^{*} をとったものとする。例えば、

  行列 A $= \begin{pmatrix} a & b \\ c & d \end{pmatrix}$の共役 $A^{\dagger}$は、$A^{\dagger} = \begin{pmatrix} a^{*} & c^{*} \\ b^{*} & d^{*} \end{pmatrix}$ である。

  行列 B $= \begin{pmatrix} 1 & 2-i \\ 3+i & 4 \end{pmatrix}$の共役 $B^{\dagger}$は、

  $B^{T} = \begin{pmatrix} 1 & 3+i \\ 2-i & 4 \end{pmatrix}$だから、$B^{\dagger} = \begin{pmatrix} 1 & 3-i \\ 2+i & 4 \end{pmatrix}$ である。
- 行列Uは、$U^{\dagger}U = UU^{\dagger} = I$ (単位行列) のとき、ユニタリと言われる。

行列 X, Z, Hの性質

$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$、$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$、$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ とする。

$$X^\dagger = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}^\dagger = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = X$$

$$Z^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = Z$$

$$H^\dagger = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^\dagger = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = H$$

$$X^\dagger = X \ , \ Z^\dagger = Z \ , \ H^\dagger = H$$

X^2, Z^2, H^2

$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ なので、

$$X^2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I$$

$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ なので、

$$Z^2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I$$

$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ なので、

$$H^2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} = I$$

$$X^2 = Z^2 = H^2 = I$$

行列 X, Z, H はユニタリ行列である

$X^\dagger = X$, $Z^\dagger = Z$, $H^\dagger = H$ と $X^2 = Z^2 = H^2 = I$ より

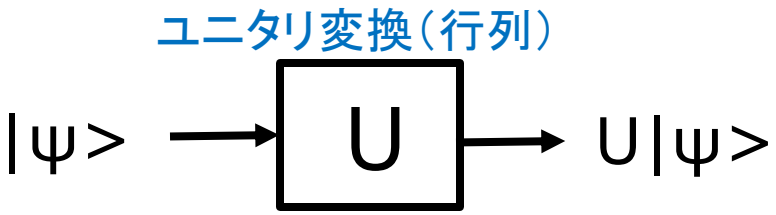
$$X^\dagger X = X X^\dagger = X^2 = \mathrm{I}$$
$$Z^\dagger Z = Z Z^\dagger = Z2 = \mathrm{I}$$
$$H^\dagger H = H H^\dagger = H^2 = \mathrm{I}$$

行列 X, Z, H はユニタリ行列である

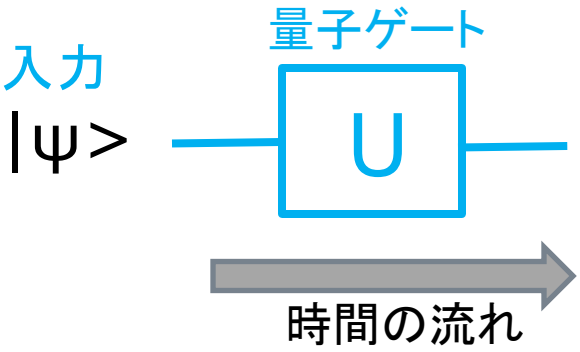
量子ゲート ＝ ユニタリ行列

量子ゲート ＝ ユニタリ行列

- 量子の状態 $|\psi\rangle$は、ユニタリ変換U（ユニタリ行列）の作用を受けて、状態 $U|\psi\rangle$ に変化する。
- この変化 $|\psi\rangle \rightarrow U|\psi\rangle$を、次のように表そう。

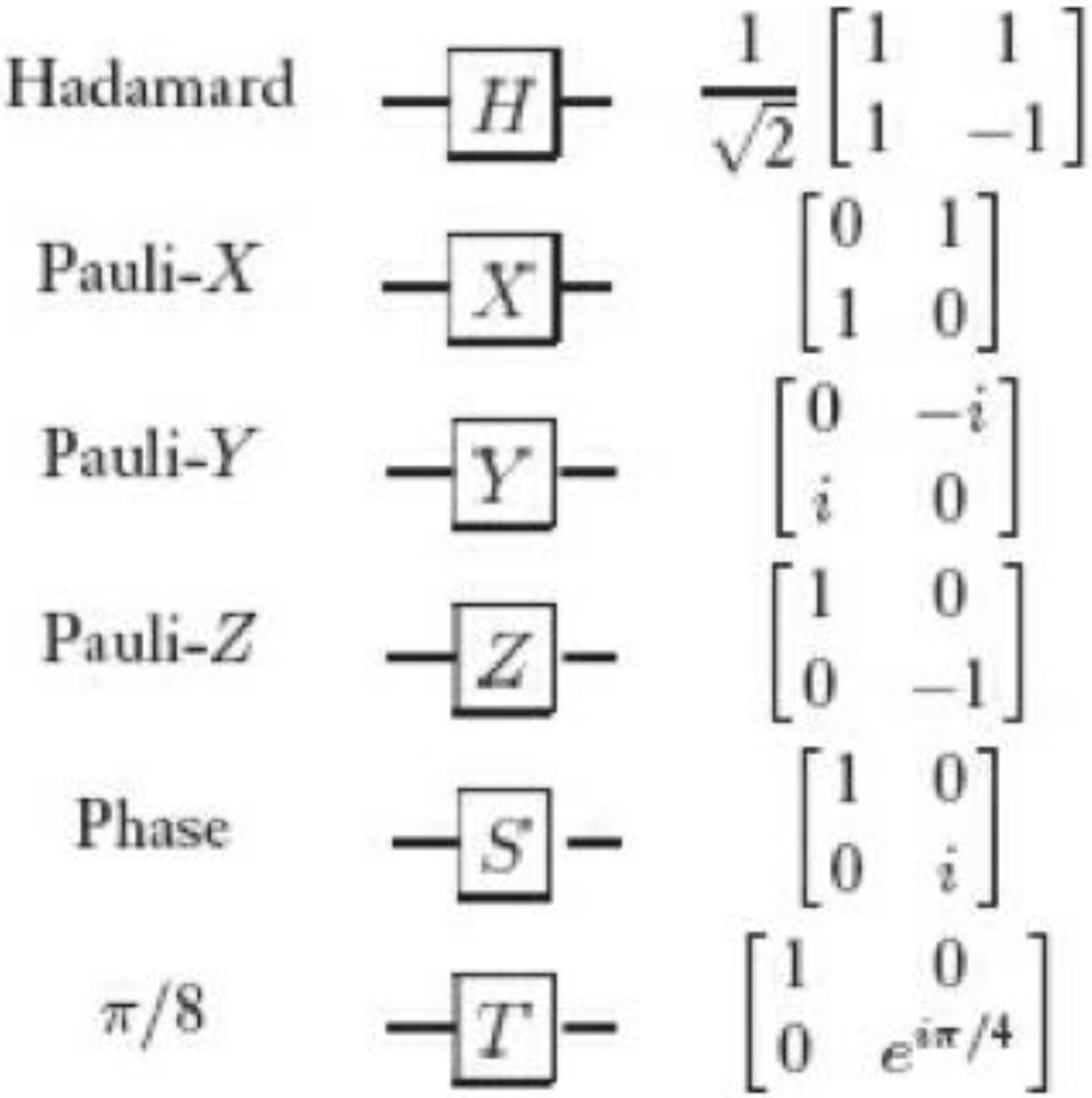


- この時、Uを、$|\psi\rangle$ を入力、$U|\psi\rangle$ を出力とする回路と考えることができる。これを、「量子ゲート」と呼ぶ。



量子ゲートは
ユニタリ行列と
一対一に対応する

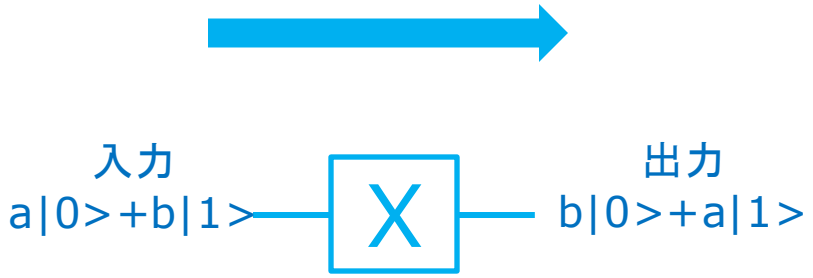
1-qubitの量子ゲートの例とそれに対応する行列



X, Z, H の行列と対応する量子ゲートの働き

$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ なので、

$$X\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} b \\ a \end{pmatrix}$$

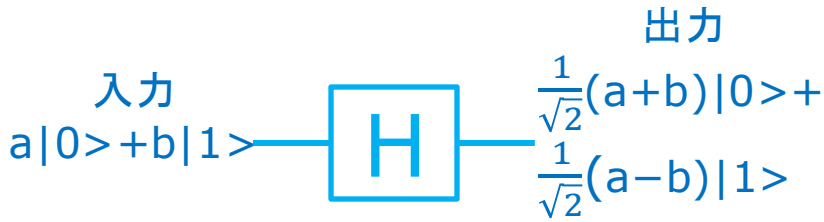


$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ なので、

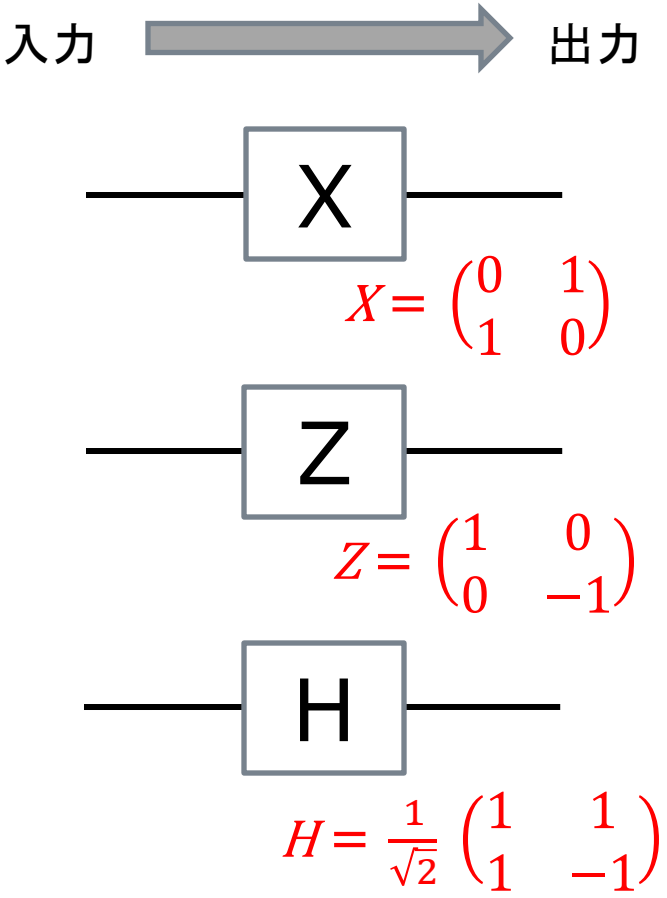
$$Z\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a \\ -b \end{pmatrix}$$

$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ なので、

$$H\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix} a+b \\ a-b \end{pmatrix}$$



代表的な量子ゲート X, Z, H
いずれもユニタリ行列で表現される



Bit Flipper

$a|0\rangle+b|1\rangle \rightarrow b|0\rangle+a|1\rangle$

Phase Flipper

$a|0\rangle+b|1\rangle \rightarrow a|0\rangle-b|1\rangle$

Hadamard

$a|0\rangle+b|1\rangle \rightarrow$

$\frac{1}{\sqrt{2}}(a+b)|0\rangle+\frac{1}{\sqrt{2}}(a-b)|1\rangle$

$|0\rangle,|1\rangle$基底から
$|+\rangle,|-\rangle$基底への変換

Bit Flipper X と Phase Flipper Z

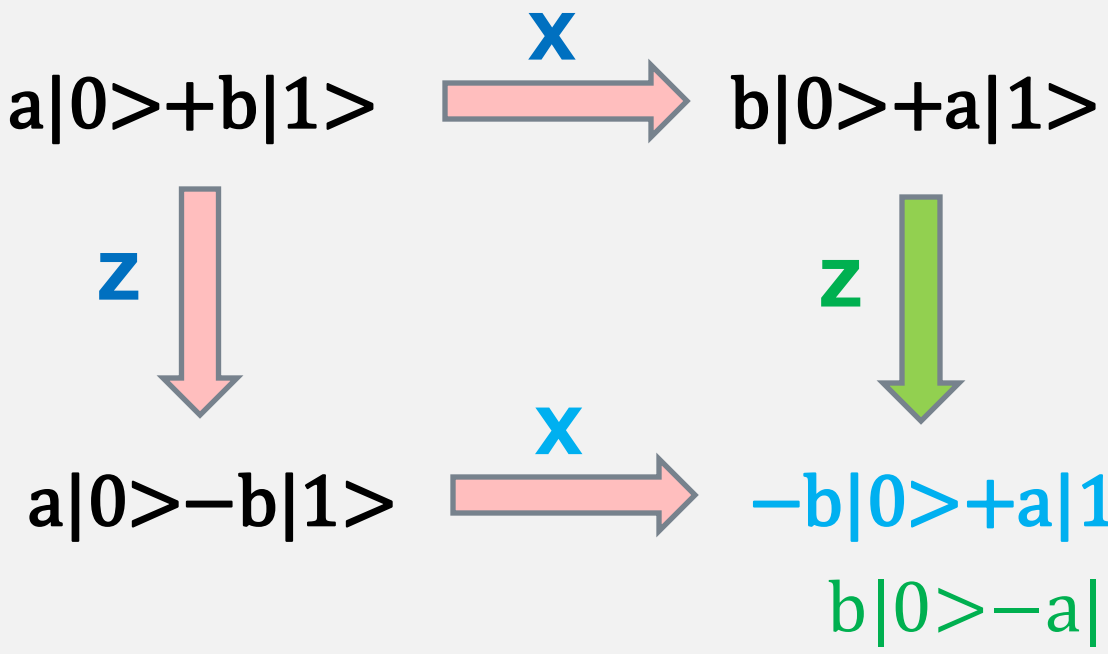
$X(a|0\rangle+b|1\rangle) = b|0\rangle+a|1\rangle$

$Z(a|0\rangle+b|1\rangle) = a|0\rangle-b|1\rangle$

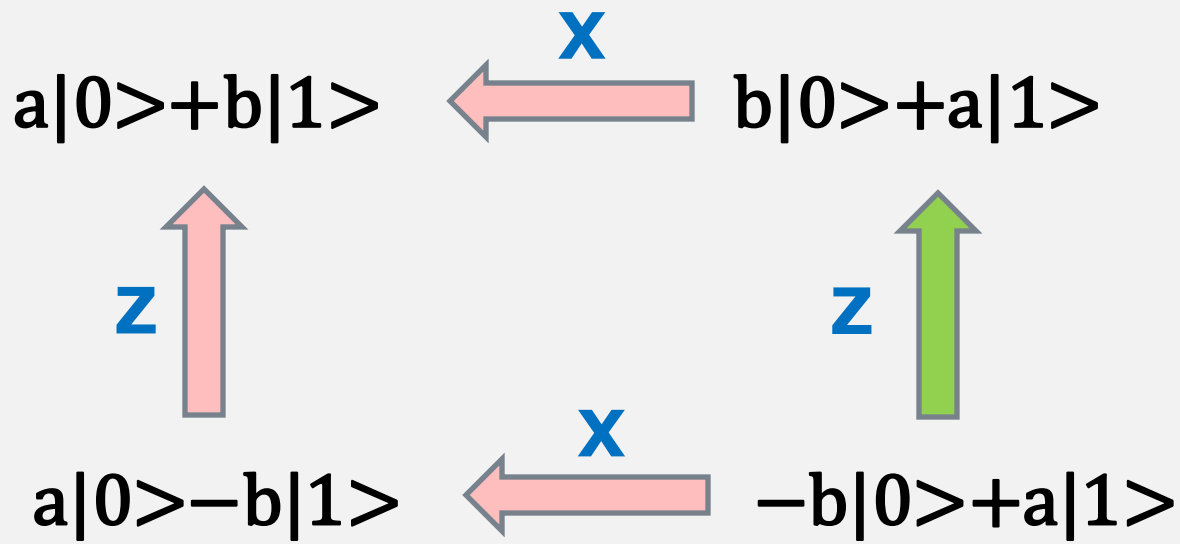
$XZ(a|0\rangle+b|1\rangle) = X(a|0\rangle-b|1\rangle) = -b|0\rangle+a|1\rangle$

$ZX(a|0\rangle+b|1\rangle) = Z(b|0\rangle+a|1\rangle) = b|0\rangle-a|1\rangle$

XZとZXは符号が逆になる



Bit Flipper X と Phase Flipper Z



$X^2 = Z^2 = I$

$X(b|0\rangle+a|1\rangle) = a|0\rangle+b|1\rangle$

$Z(a|0\rangle-b|1\rangle) = a|0\rangle+b|1\rangle$

$ZX(-b|0\rangle+a|1\rangle) = Z(a|0\rangle-b|1\rangle) = a|0\rangle+b|1\rangle$

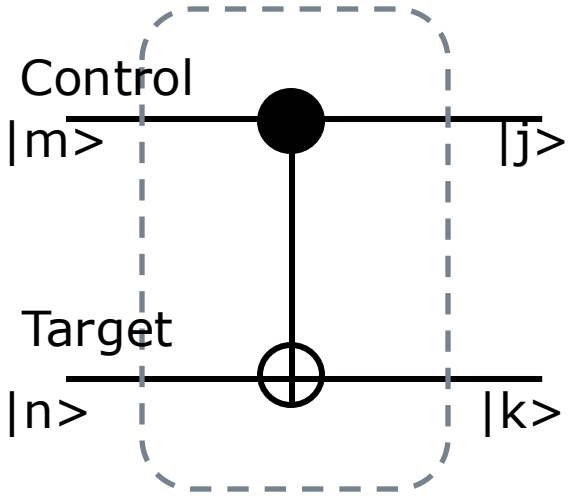
$XZ(-b|0\rangle+a|1\rangle) = X(-b|0\rangle-a|1\rangle) = -a|0\rangle-b|1\rangle$

XZとZXは符号が逆になる

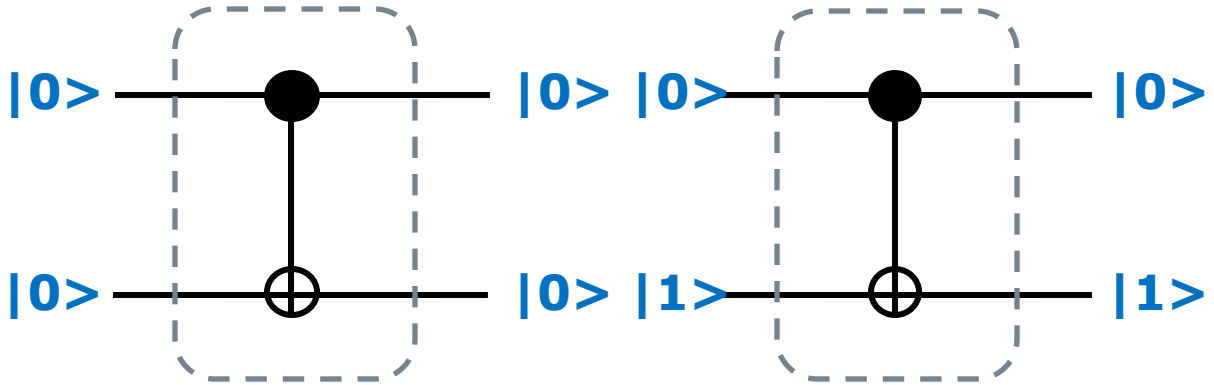
2-qubitのゲート
CNOTゲートとControl-Uゲート

2-qubitのゲート
CNOT (Control-NOT)

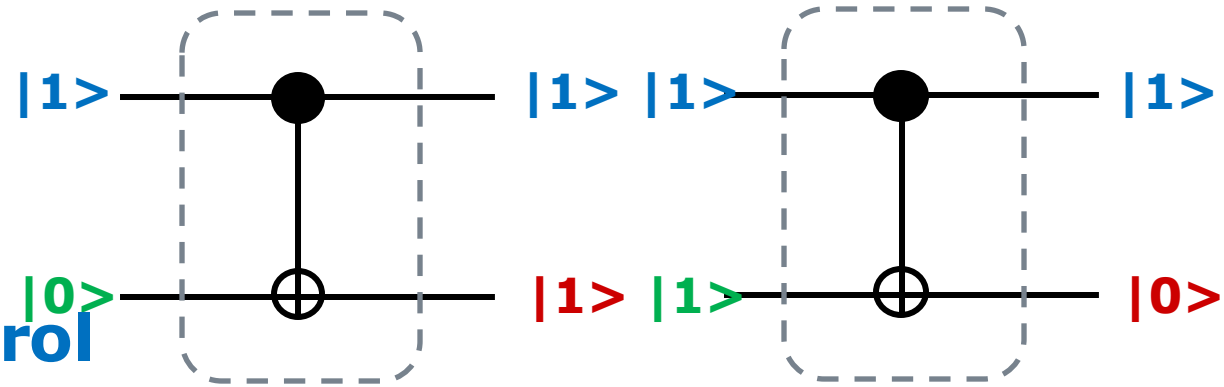
Control が |1> の時
Target のNOTをとる



Controlが |0>なら何もしない



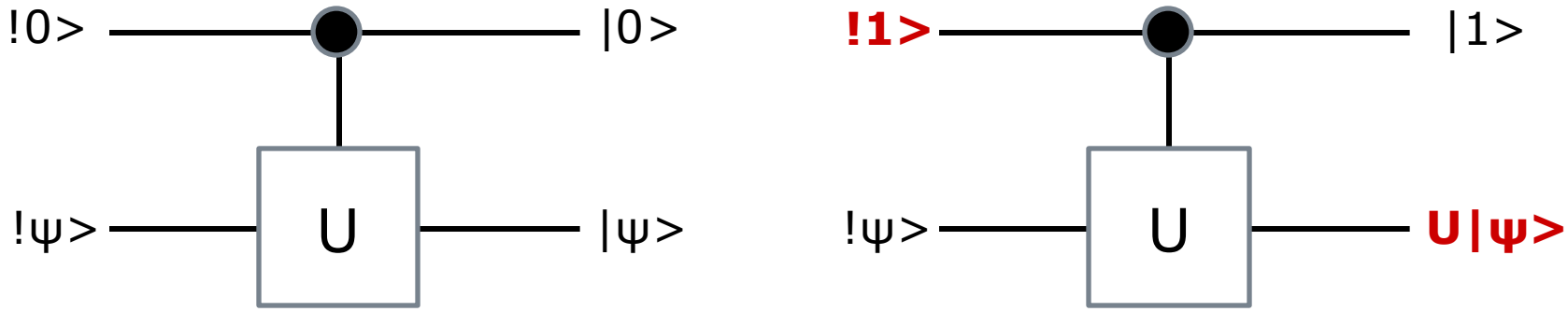
Controlが |1>ならNOT操作



|mn>の
第一bit |m>がControl
第二bit |n>がTarget

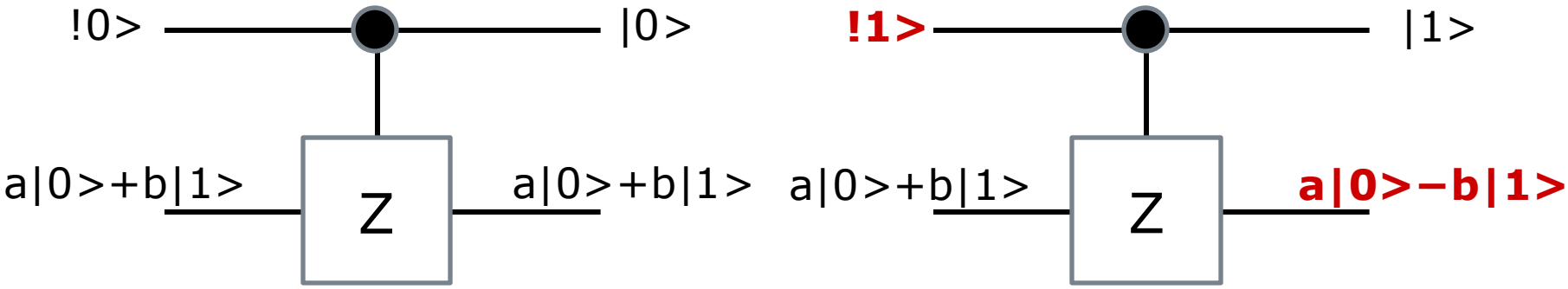
2-qubitのゲート **Control-U**

コントロール qubit が、|0> の時には、何もせず、（左図）
コントロール qubit が、|1> の時には、ターゲットのqubit |ψ> にユニタリ演算子 U を適用した U|ψ>を出力する（右図）
ゲートを、**Control-U** ゲートという。



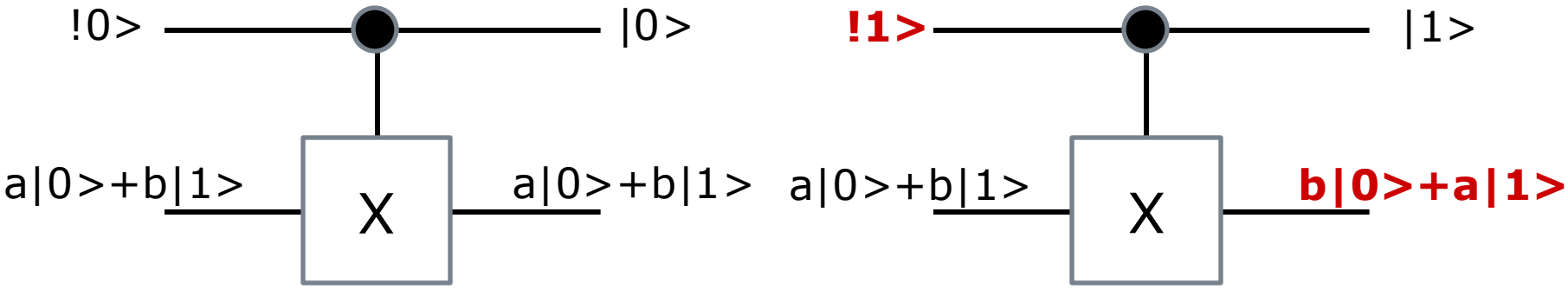
2-qubitのゲート Control-U の例 (1)
Control-Z

コントロール qubit が、|0> の時には、何もせず、(左図)
コントロール qubit が、|1> の時には、ターゲットのqubit
|ψ> にユニタリ演算子 Z を適用した Z|ψ>を出力する(右図)



2-qubitのゲート Control-U の例 (2)
Control-X

コントロール qubit が、|0> の時には、何もせず、(左図)
コントロール qubit が、|1> の時には、ターゲットのqubit
|ψ> にユニタリ演算子 X を適用した X|ψ>を出力する(右図)



Control-Xは、CNOTと等しいことを確かめよ

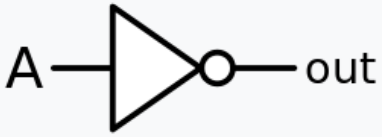
古典ゲートと量子ゲート

古典ゲートと量子ゲートの比較

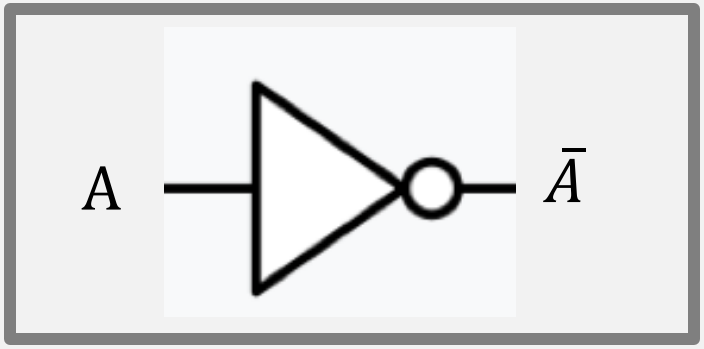
- ここでは、まず、古典的ゲートと量子ゲートの共通点を持て見よう。もちろん、入力があって出力があるのは同じである。
- qubitの|0>,|1> をそれぞれ古典ビットの 0, 1 に等しいとみなすと、入力と出力の値の対応を見れば、よく似た働きをするゲートを見つけることはできる。例えば、
  - 1bitのNOTゲートと1qubitのXゲート
  - 2bitのXORゲートと2qubitのCNOTゲート

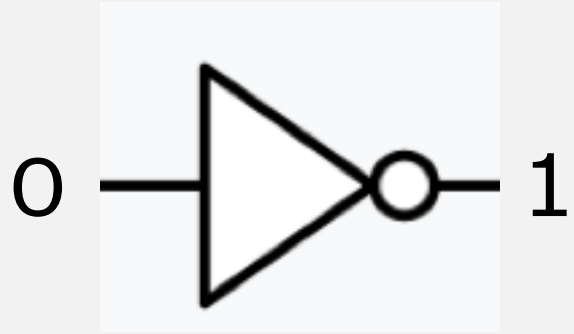
- ただ、この例でも、XORゲートの出力は一つだが、CNOTの出力は二つである。実は、量子ゲートでは、入力の数と出力の数は、つねに同じでなければならない。これについては後で述べる。

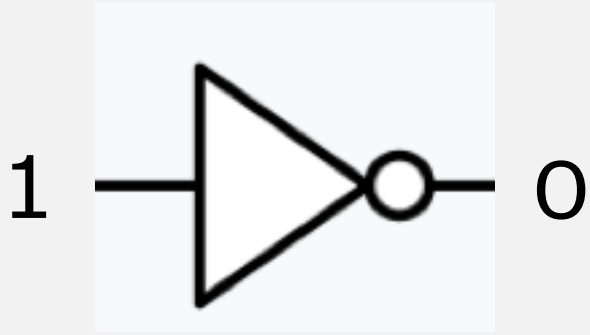
古典論理ゲートの例

論理	論理式	回路記号（MIL記号）
NOT	$\overline{A}$	
OR	$A + B$	
AND	$A \cdot B$	
XOR	$A \oplus B$	

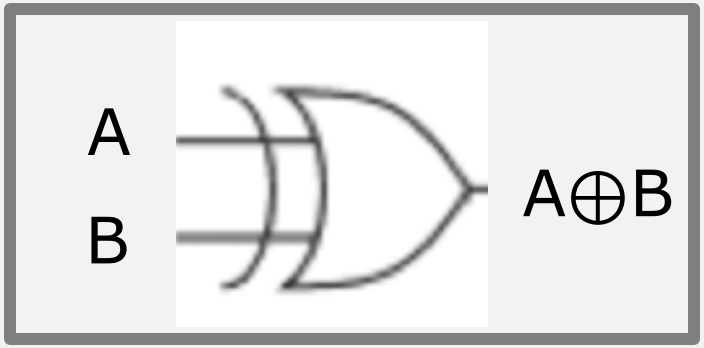
否定の論理ゲート（bit)

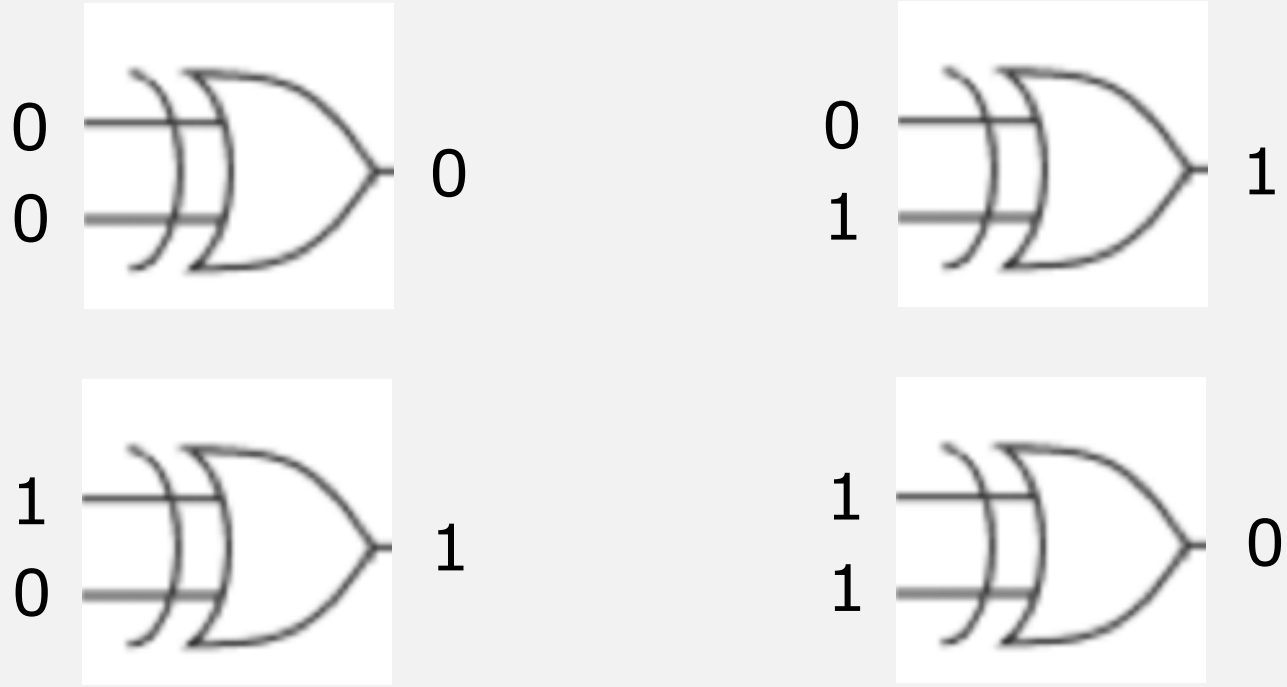




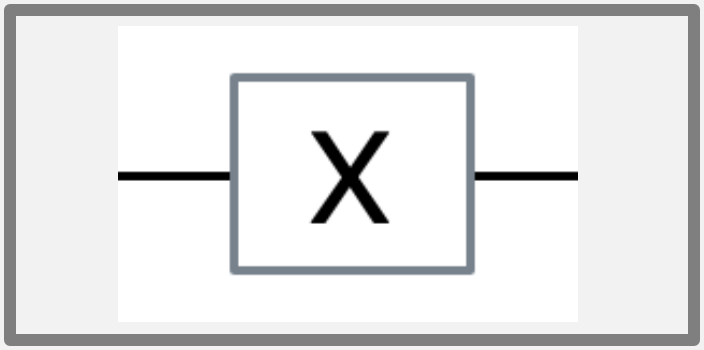


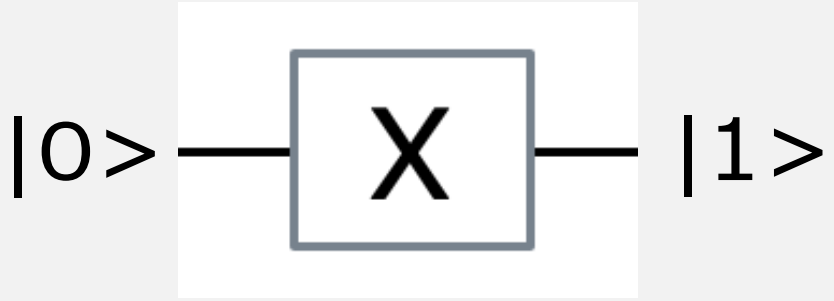
排他的論理和の論理ゲート（bit)

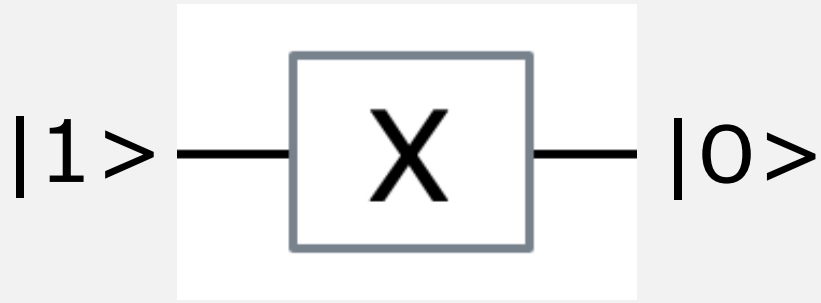




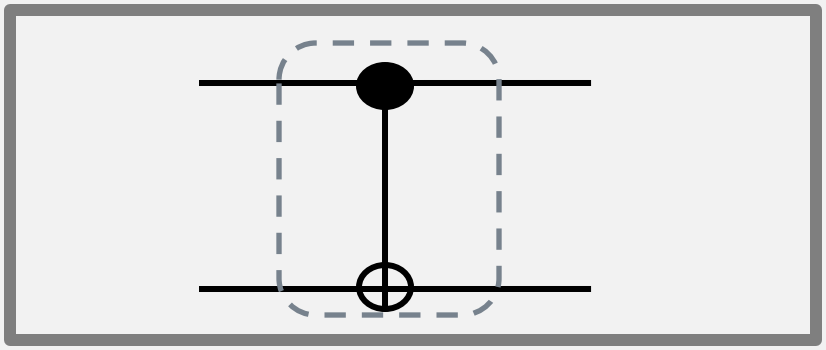
量子ゲート X（qubit)

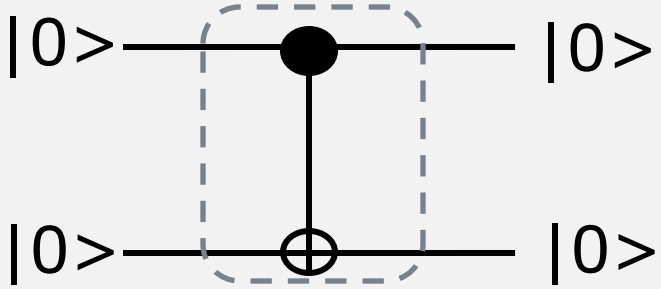


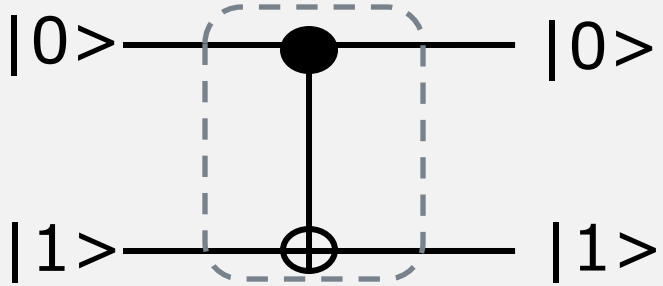


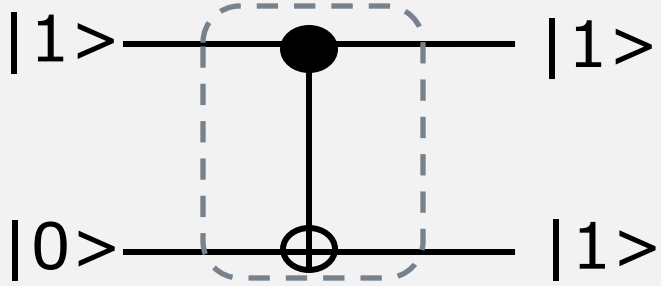


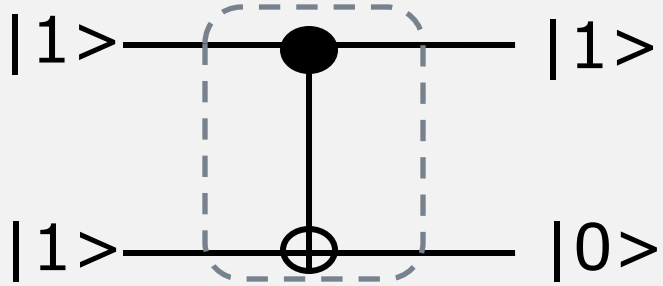
CNOT量子ゲート (qubit)



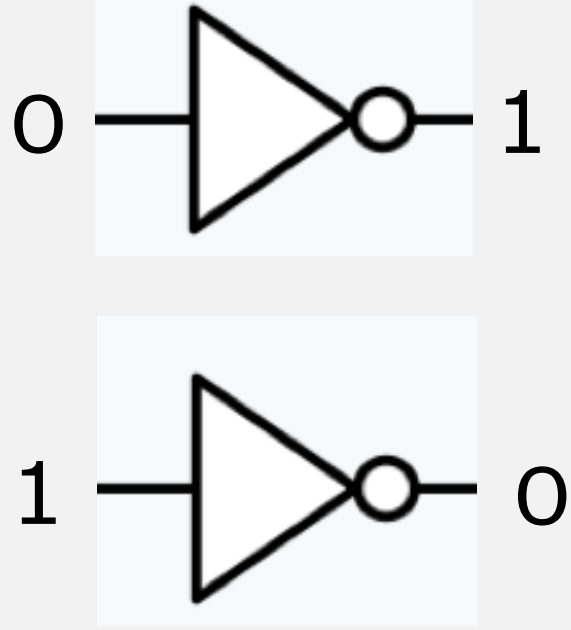






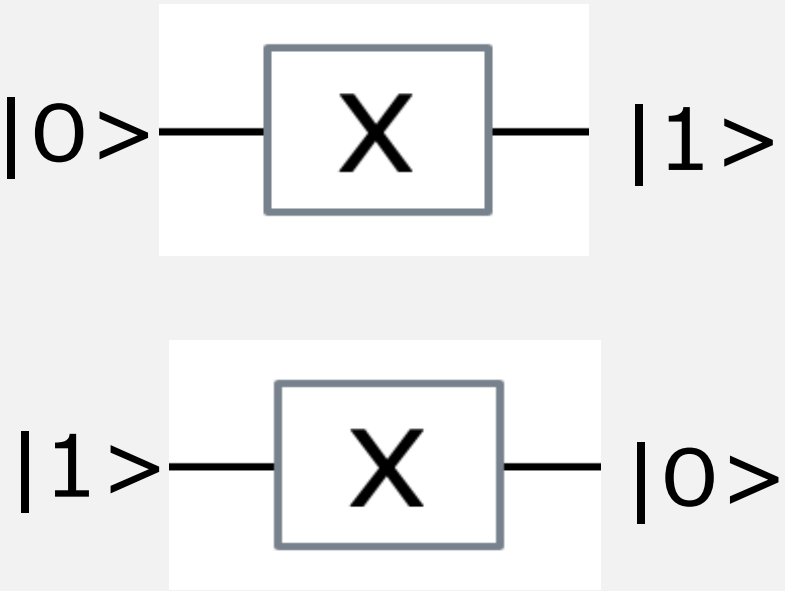


bit

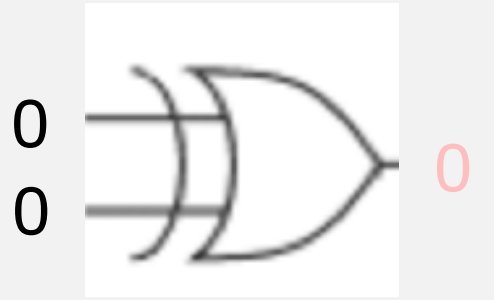


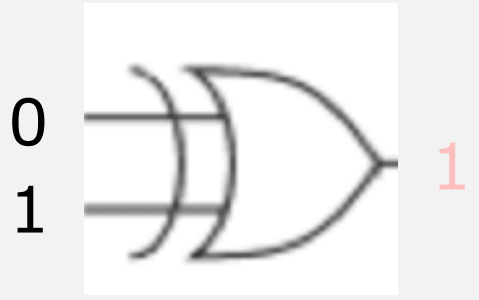
二つのゲートは、1対応している

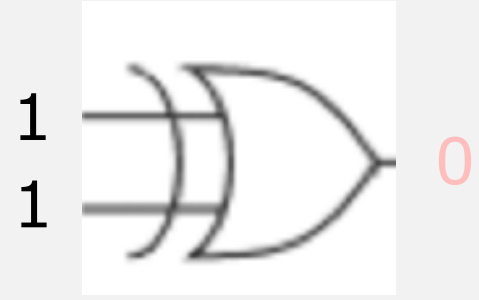
qubit



bit

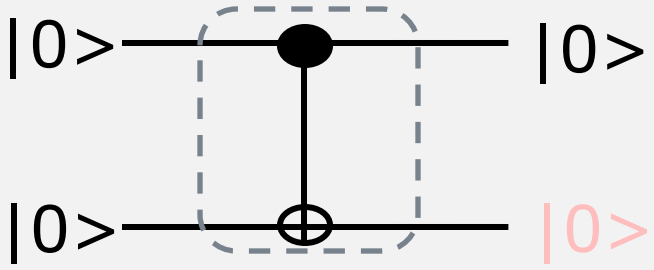


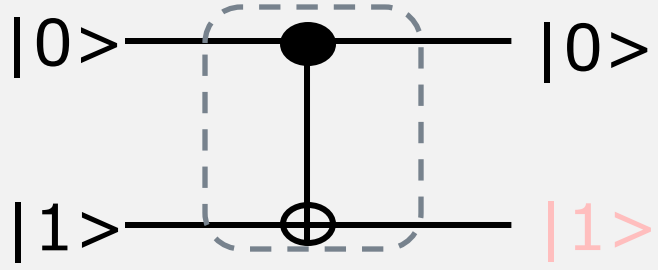


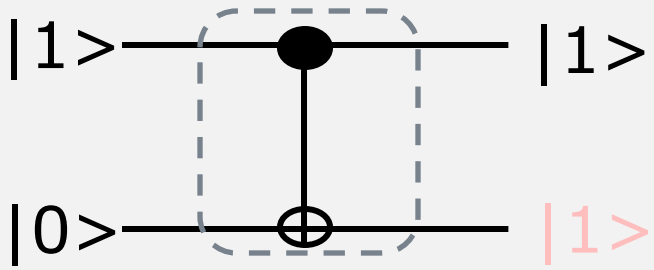


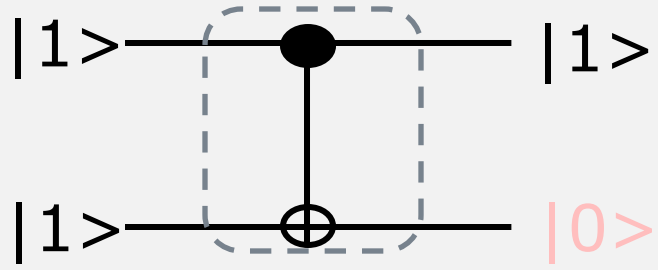
二つのゲートは、1対応している

qubit









量子回路

-- 複数の量子の状態をテンソルで表す --

量子回路
-- 複数の量子の状態をテンソルで表す --

- 複数の量子ゲートをつなげて量子回路を作る
- 複数のゲートを直列に組み合わせる
- 複数のレジスターを並列に組み合わせる
- 複数のレジスターのテンソル積の例
- 二つの1-qubitのゲートを、並列に組み合わせる
- テンソル積の定義
- 2-qubitの基底

複数の量子ゲートをつなげて量子回路を作る

- 複数の量子ゲートをつなげて量子回路を作ることができる。
  基本的には、次の二つのパターンでゲートを配置する。
  - ゲートを直列に配置する。
  - ゲートを並列に配置する。

複数の量子ゲートをつなげて量子回路を作る

- 1-qubitのゲートを直列につなげて構成された量子回路の働きを理解するのは、比較的容易である。
  - 入力に対して、ゲートに対応する行列を順番にかけていけば、出力は計算できる。（行列の積を計算する）
  - このとき、構成されるラインをレジスターということがある。レジスターの値は、ゲートを通過するたびに変化すると考えればいい。

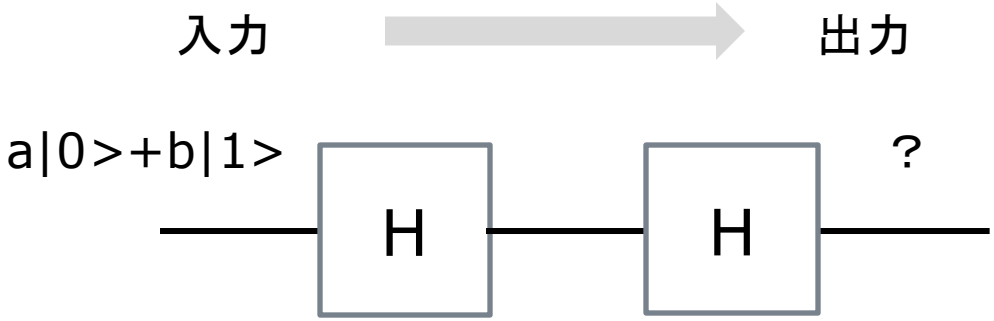
- ゲートが並列に配置されている場合は、少し複雑である。
  - 基本的には、回路を構成する複数のパラレルなレジスターのテンソル積を計算することになる

ゲートを直列に組み合わせる

ゲートを直列に組み合わせる

ゲートを直列に組み合わせた回路の働きは、直列の回路を
構成するゲートの行列の積を計算することで、求めることができる。

二つのHゲートを直列につないだ、次のような回路を考えてみよう。

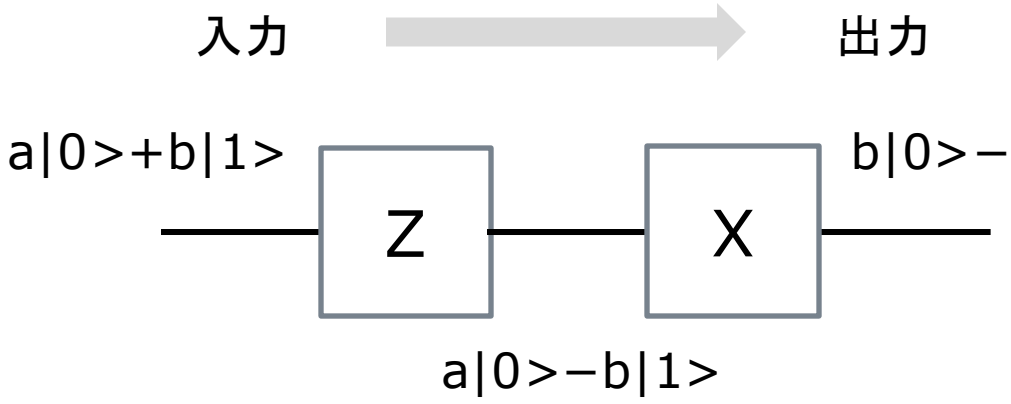


$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ なので、行列の積HHを計算する。

$$HH = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

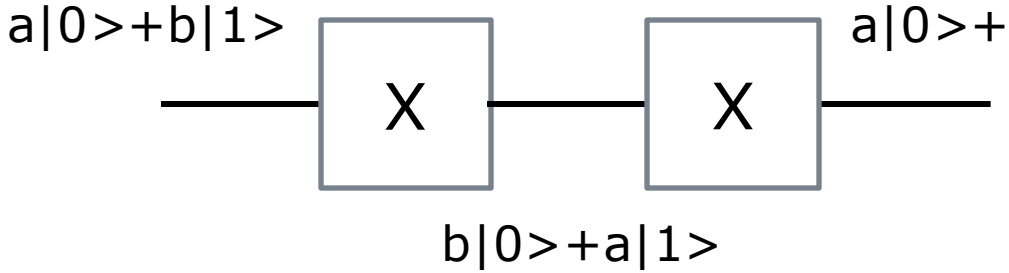
これは単位行列なので、先の回路の出力は、a|0>+b|1>

1-qubitのゲートを、直列に組み合わせる (1)



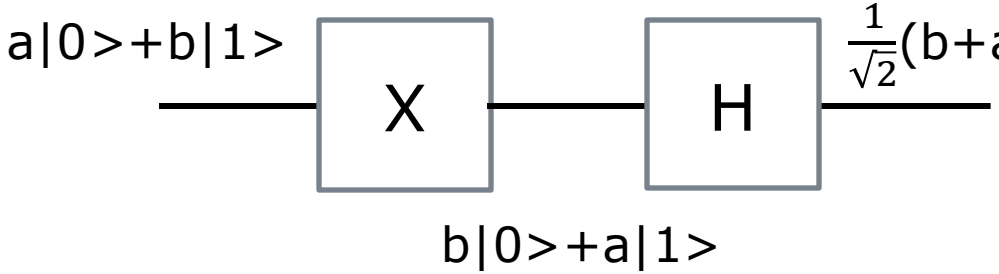
$$= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

X　　Z



$$= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

X　　X



1-qubitのゲートを、直列に組み合わせる (2)

$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}$ とする。

$$R_1 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

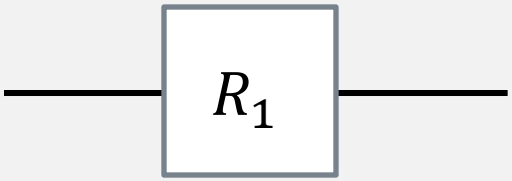
$$R_2 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

$$R_3 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^3} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} i \end{pmatrix}$$

$$R_4 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^4} \end{pmatrix} = \cdots$$

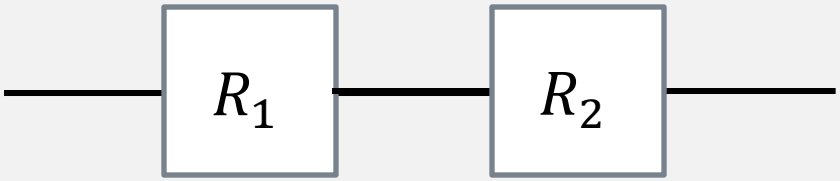
1-qubitのゲートを、直列に組み合わせる (2)

入力 → 出力



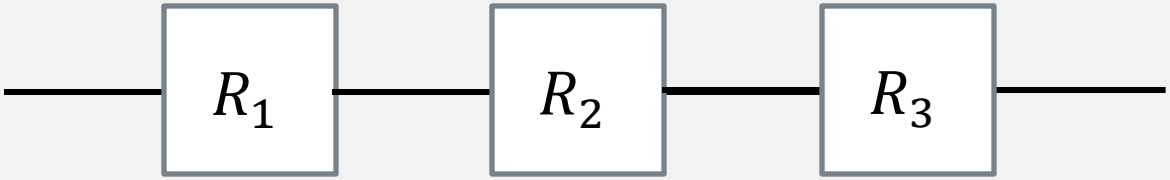
$$\begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^1} \end{pmatrix} |\psi>$$

R_1



$$\begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^2} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^1} \end{pmatrix} |\psi>$$

R_2 R_1



$$\begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^3} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^2} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^1} \end{pmatrix} |\psi>$$

R_3 R_2 R_1

複数のレジスターを
並列に組み合わせる

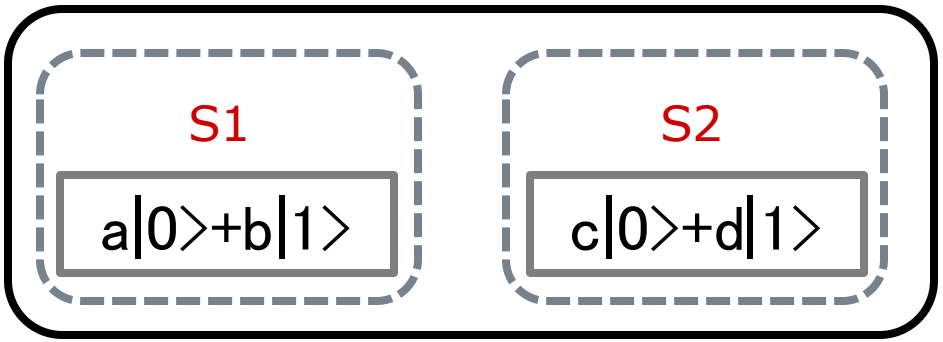
テンソル積
独立した二つのシステムを一つのシステムと考える

qubit a|0>+b|1> のみを含むシステムをS1、

qubit c|0>+d|1> のみを含むシステムをS2とした時、

この二つを一緒にしたシステムSを テンソル積S1⊗ S2で表す。

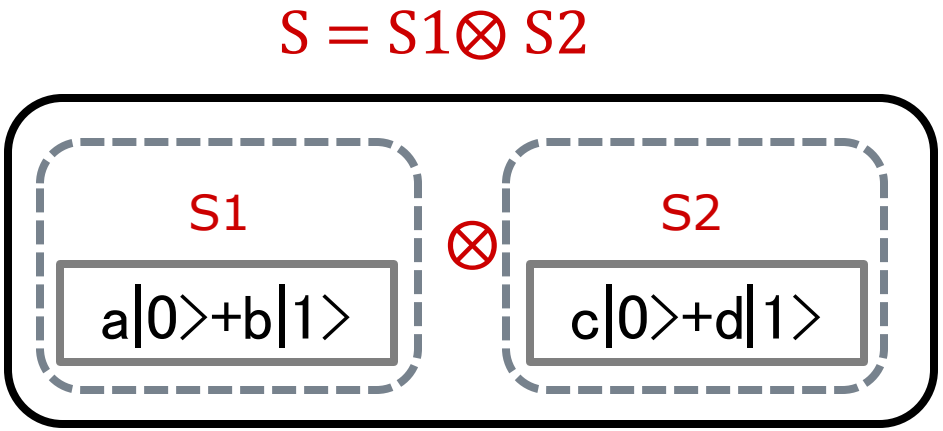
$$S = S1 \otimes S2$$



2つのqubitからなるシステム （テンソル積）

この時、次のような計算で、Sの状態を計算する。

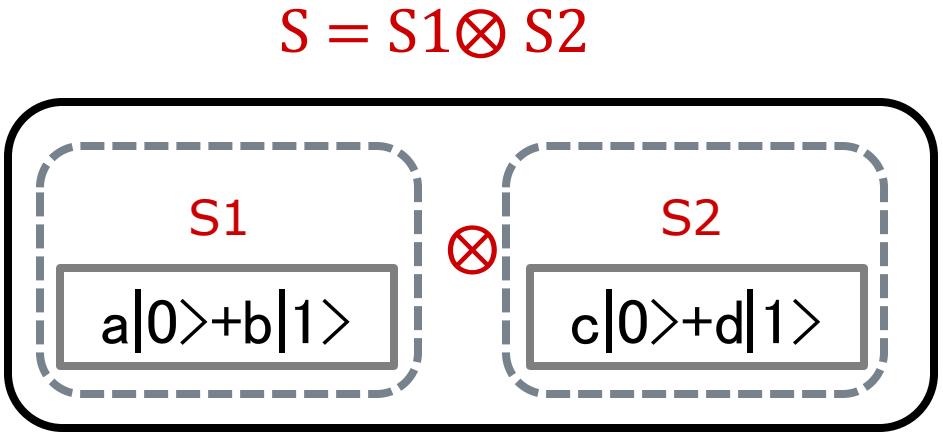
$$S = S1 \otimes S2 = (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle)$$
$$= ac|0\rangle\otimes|0\rangle + ad|0\rangle\otimes|1\rangle + bc|1\rangle\otimes|0\rangle + bd|1\rangle\otimes|1\rangle$$



2つのqubitからなるシステム （テンソル積）

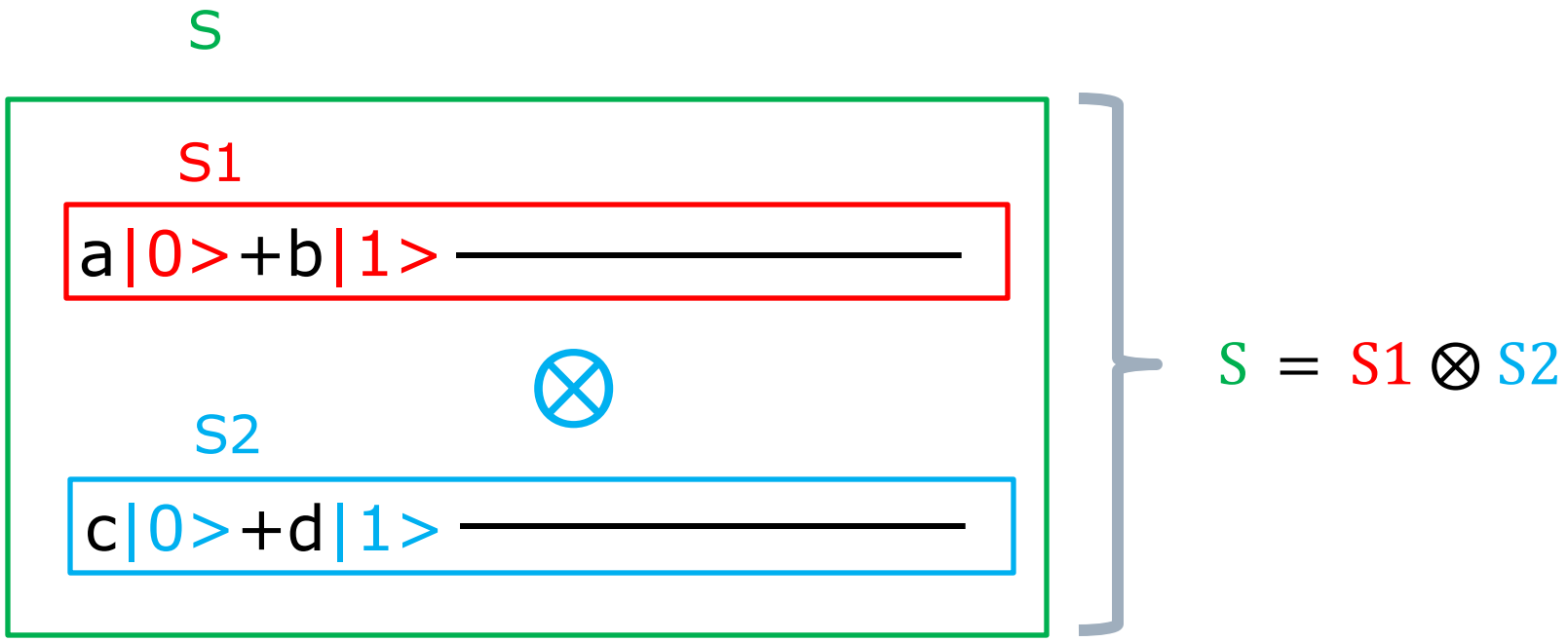
|x>⊗|y>を |xy>と表すことにすると、

$$S = S1 \otimes S2 = (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle)$$
$$= ac|0\rangle\otimes|0\rangle + ad|0\rangle\otimes|1\rangle + bc|1\rangle\otimes|0\rangle + bd|1\rangle\otimes|1\rangle$$
$$= ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle$$



2つのqubit S1, S2からなるシステム S

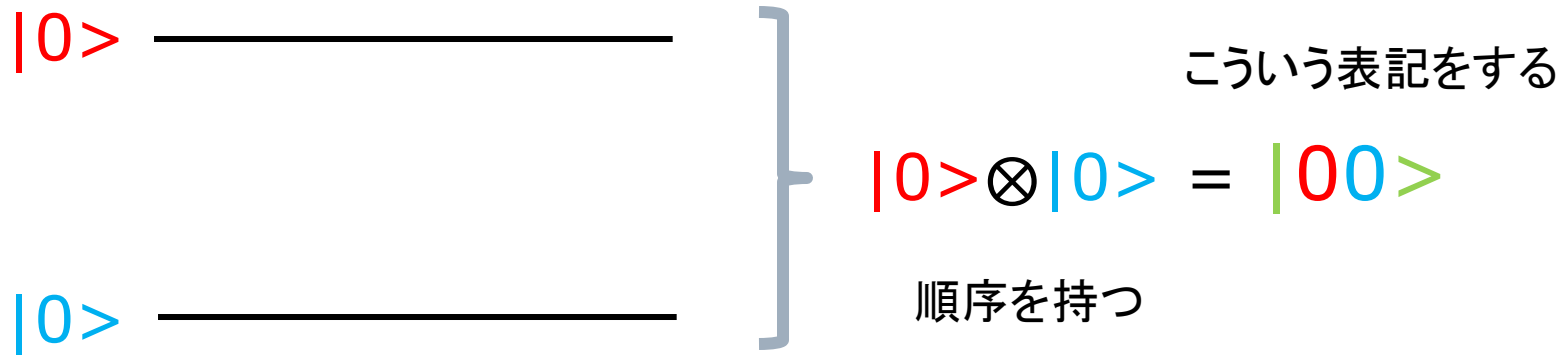
$$S = S1 \otimes S2$$



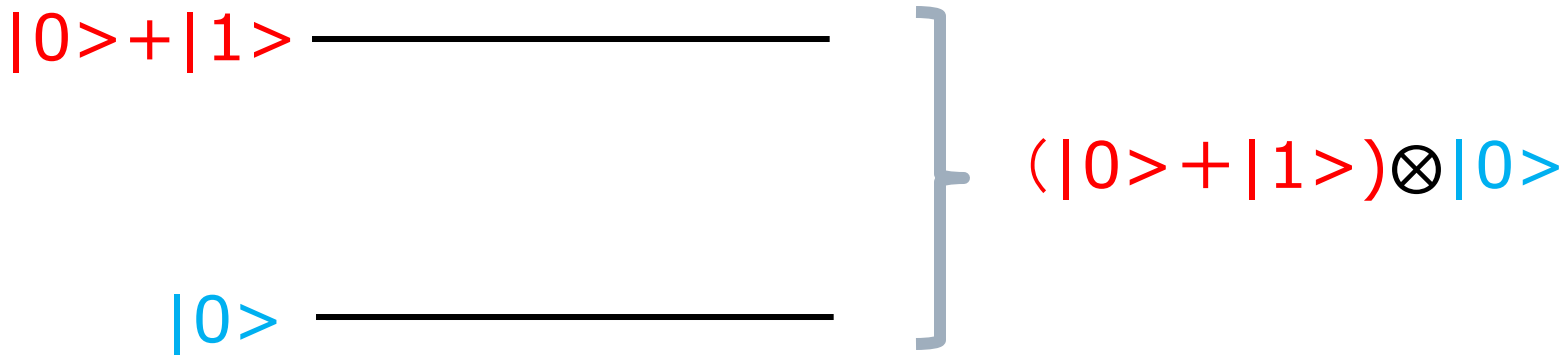
$$S = S1 \otimes S2 = (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle)$$
$$= ac|0\rangle\otimes|0\rangle + ad|0\rangle\otimes|1\rangle + bc|1\rangle\otimes|0\rangle + bd|1\rangle\otimes|1\rangle$$
$$= ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle$$

複数のレジスターの
テンソル積の例

二つのレジスターのテンソル積の例　単純な例 1



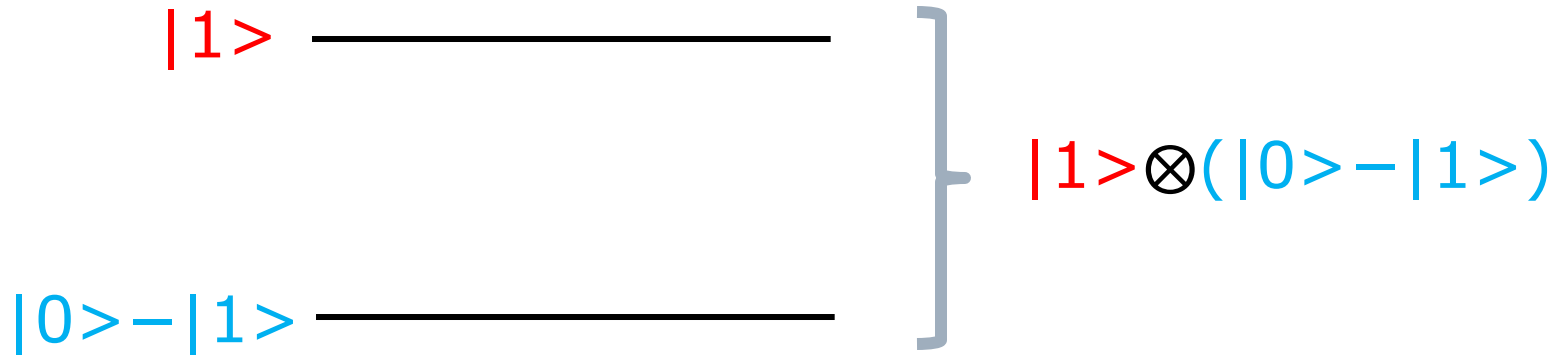
二つのレジスターのテンソル積の例　単純な例 2



$$(|0\rangle+|1\rangle)\otimes|0\rangle$$
$$=|0\rangle\otimes|0\rangle+|1\rangle\otimes|0\rangle=|00\rangle+|10\rangle$$

$|0\rangle+|1\rangle$という形は本当はおかしい。$\alpha|0\rangle+\beta|1\rangle$の時、$|\alpha|^2+|\beta|^2=1$ という条件があるからだ。本当は、$\frac{1}{\sqrt{2}}|0\rangle+\frac{1}{\sqrt{2}}|1\rangle$なのだが、計算では無視している。

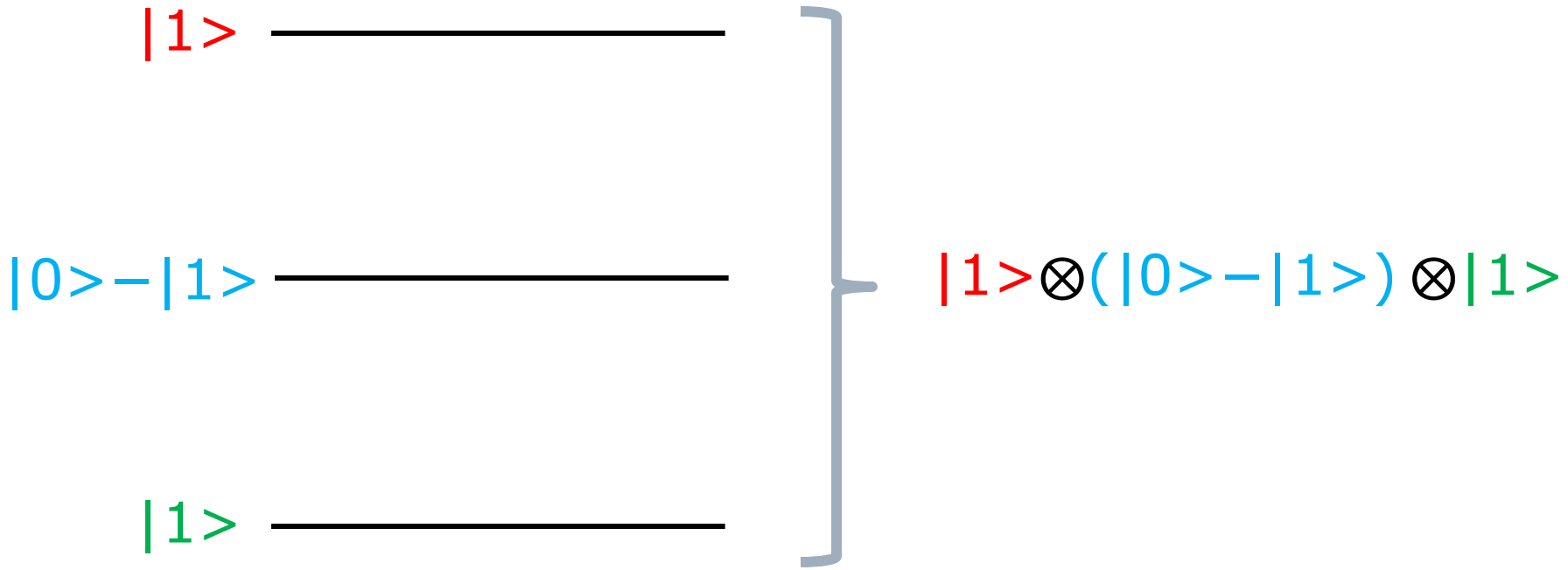
二つのレジスターのテンソル積の例　単純な例 3



$|1\rangle \otimes (|0\rangle - |1\rangle) =$
$|1\rangle \otimes |0\rangle - |1\rangle \otimes |1\rangle = |10\rangle - |11\rangle$

$|0\rangle - |1\rangle$という形は本当はおかしい。$\alpha|0\rangle + \beta|1\rangle$の時、$|\alpha|^2 + |\beta|^2 = 1$ という条件があるからだ。本当は、$\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$なのだが、計算では無視している。

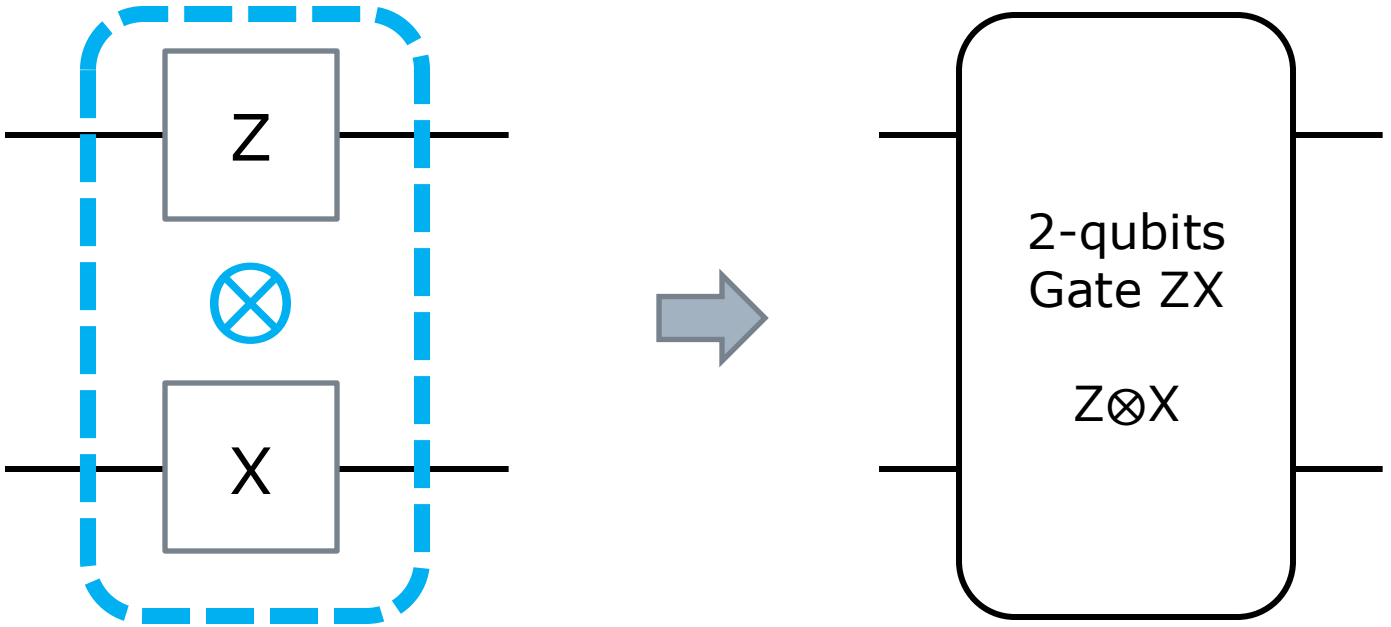
三つのレジスターのテンソル積の例　単純な例 4



$$
\begin{aligned}
&|1\rangle \otimes (|0\rangle - |1\rangle) \otimes |1\rangle \\
&= (|1\rangle \otimes |0\rangle - |1\rangle \otimes |1\rangle) \otimes |1\rangle \\
&= (|10\rangle - |11\rangle) \otimes |1\rangle = |101\rangle - |111\rangle
\end{aligned}
$$

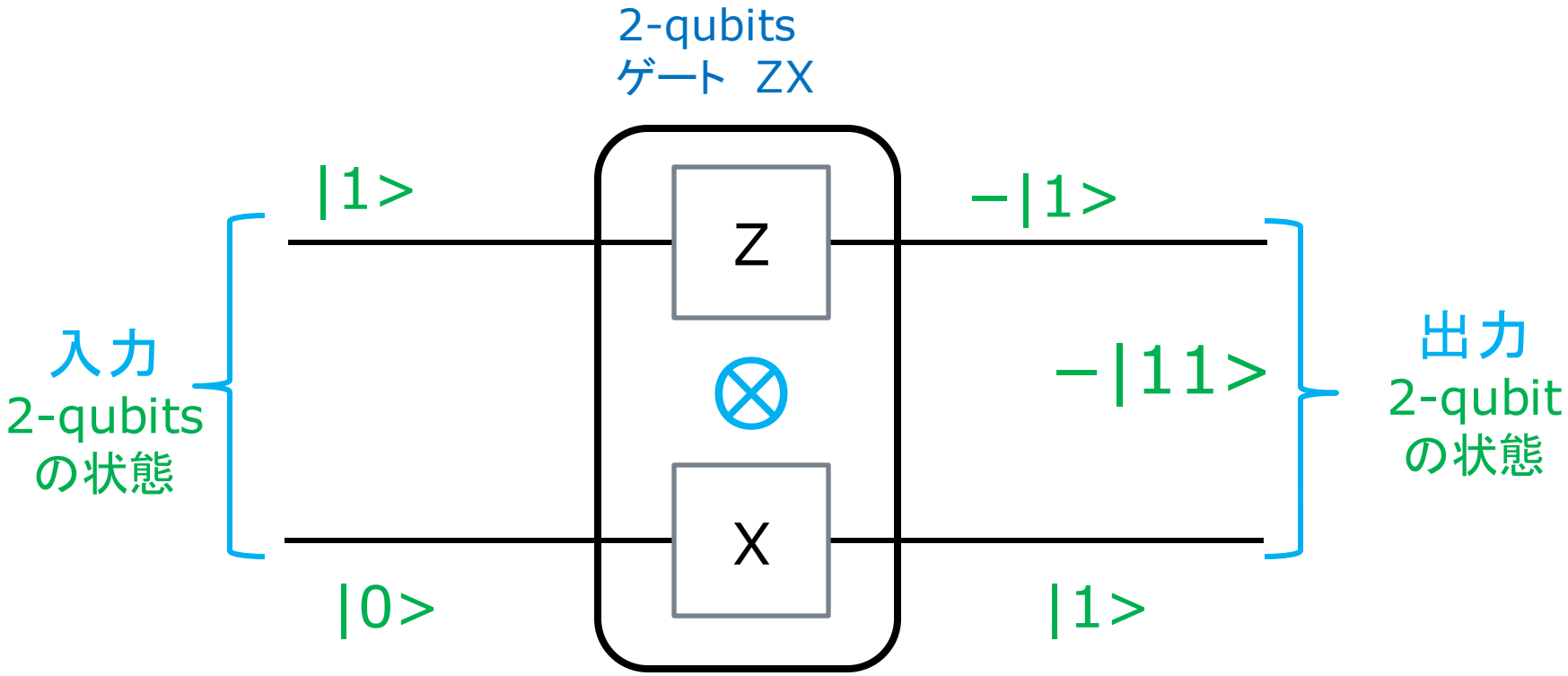
二つの1-qubitのゲートを、並列に組み合わせる

二つの 1-qubit ゲートがある時、それらを並列に組み合わせて、2-qubitsのゲートを構成することができる。二つのものを一つに考えるにはテンソル積を使えばいい。ただし、状態のテンソル積ではなく、ゲートのテンソル積である。例えば、次の2-qubitsゲートは、ZゲートとXゲートをパラレルに組み合わせたものだ。この2-qubitゲート ZX の働きを見てみよう。



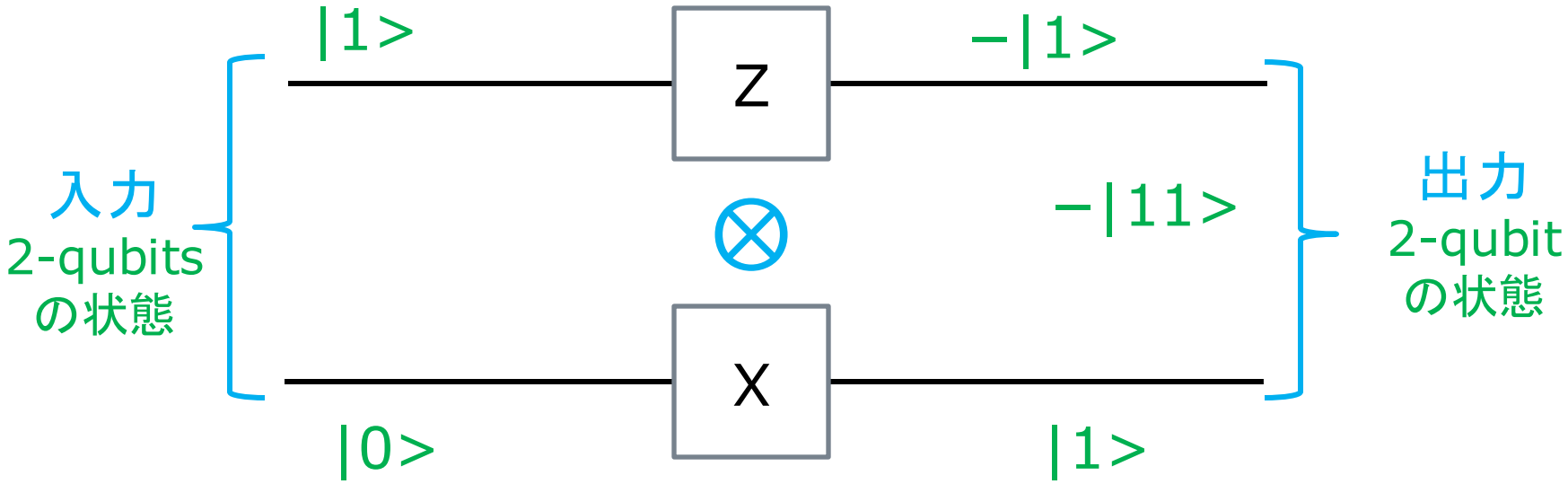
2-qubits ゲート ZX
入出力サンプル 1

- ゲートZXの働きを、いくつかの入出力の例で見ておこう。
  この結果は、2-qubitsゲートZXがなくて、二つの1-qubitゲートのパラレルに置かれたものの働きに等しい。



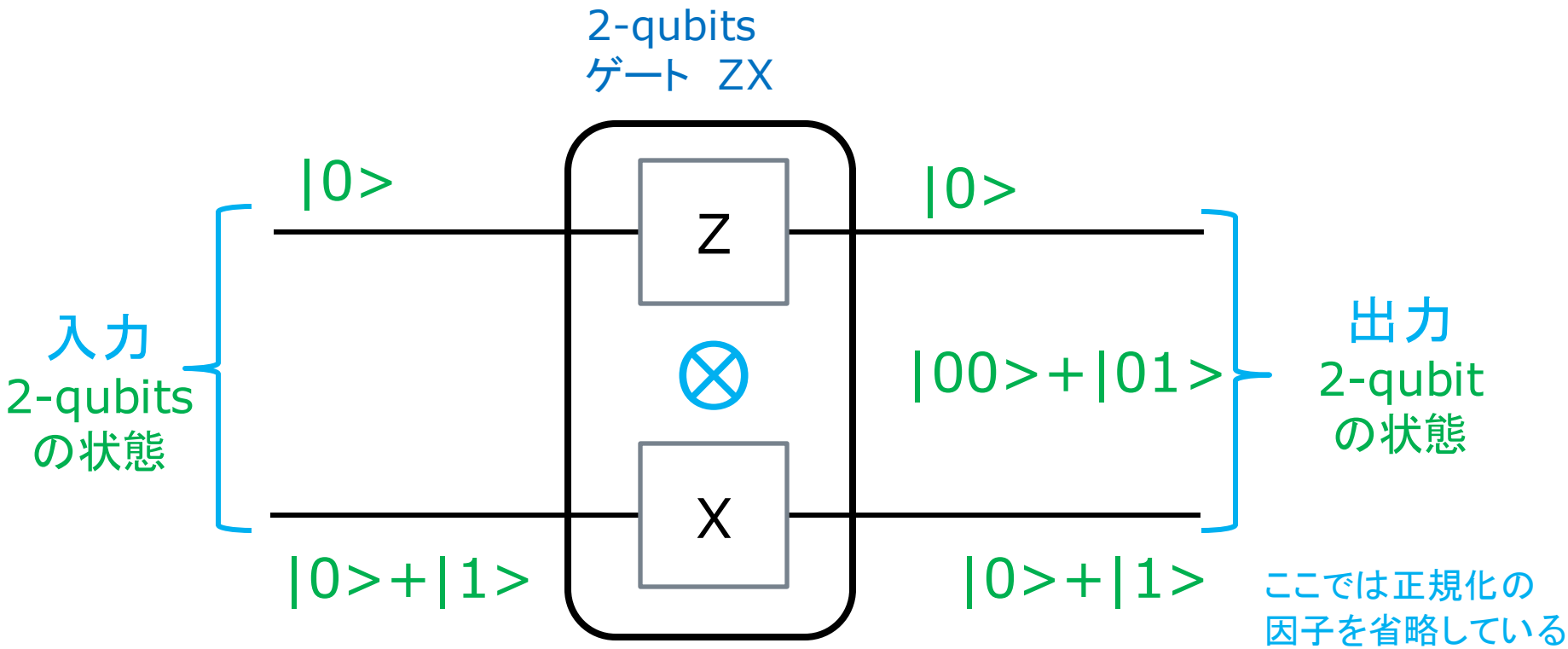
2-qubits ゲート ZX
入出力サンプル 1

- この結果は、2-qubitsゲートZXがなくて、二つの1-qubitゲート ZとXがパラレルに置かれたものの働きに等しい。



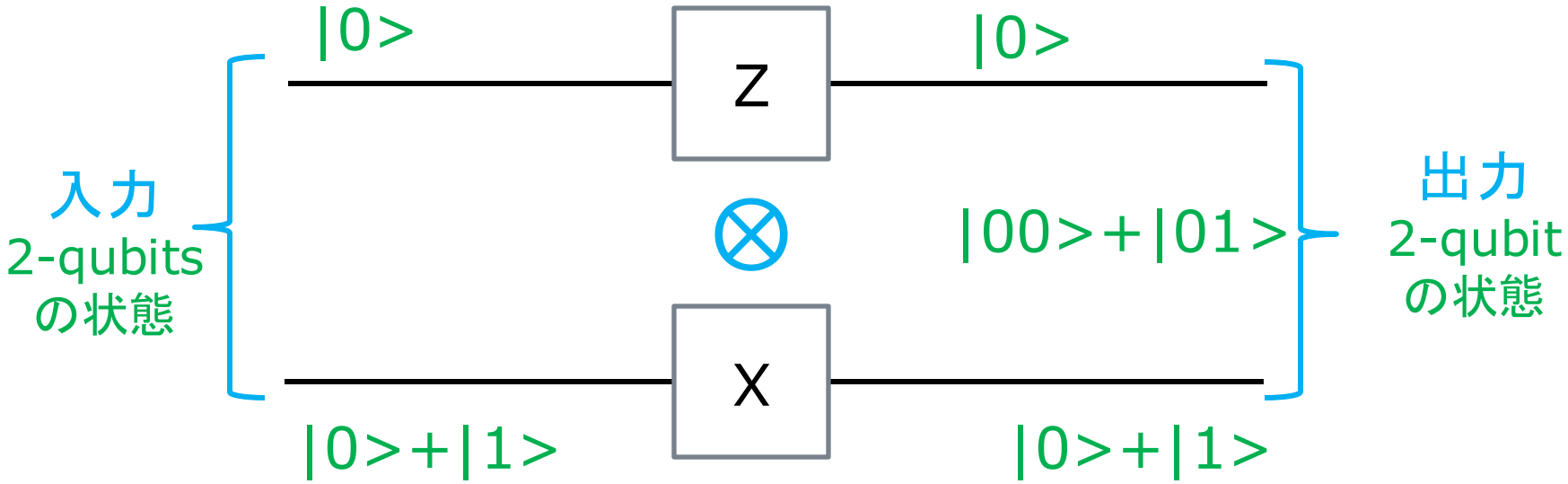
2-qubits ゲート ZX
入出力サンプル 2

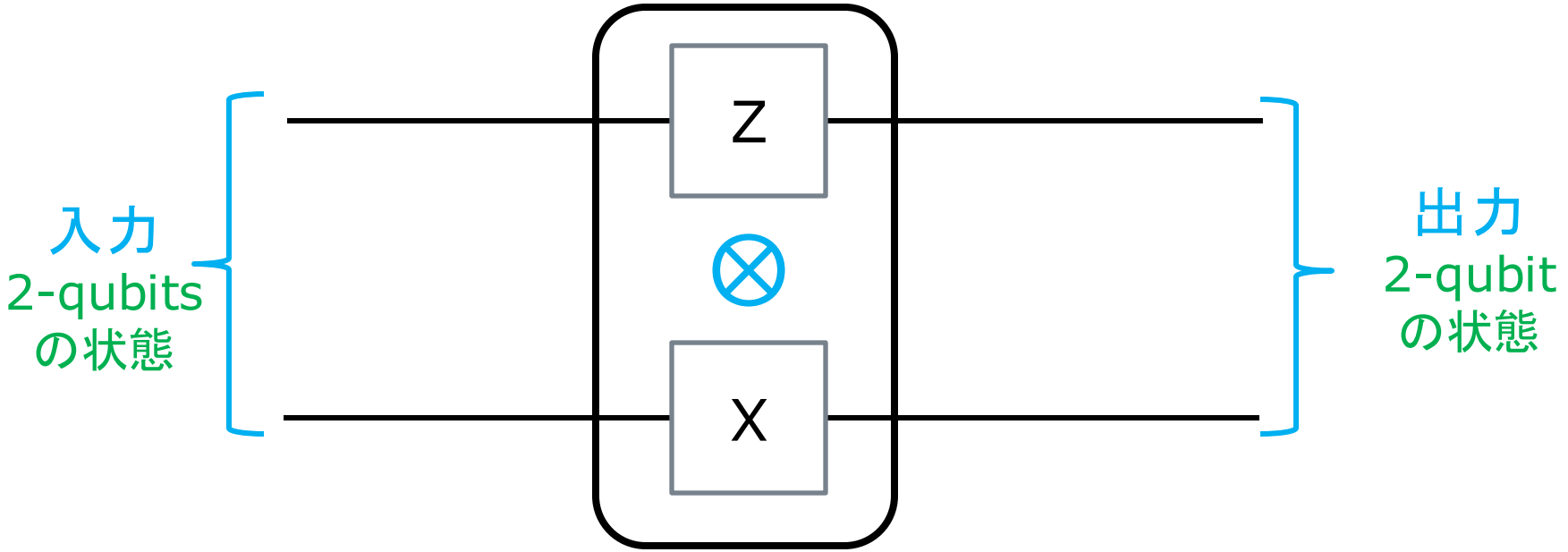
- ゲートZXの働きを、いくつかの入出力の例で見ておこう。
  この結果は、2-qubitsゲートZXがなくて、二つの1-qubitゲートのパラレルに置かれたものの働きに等しい。

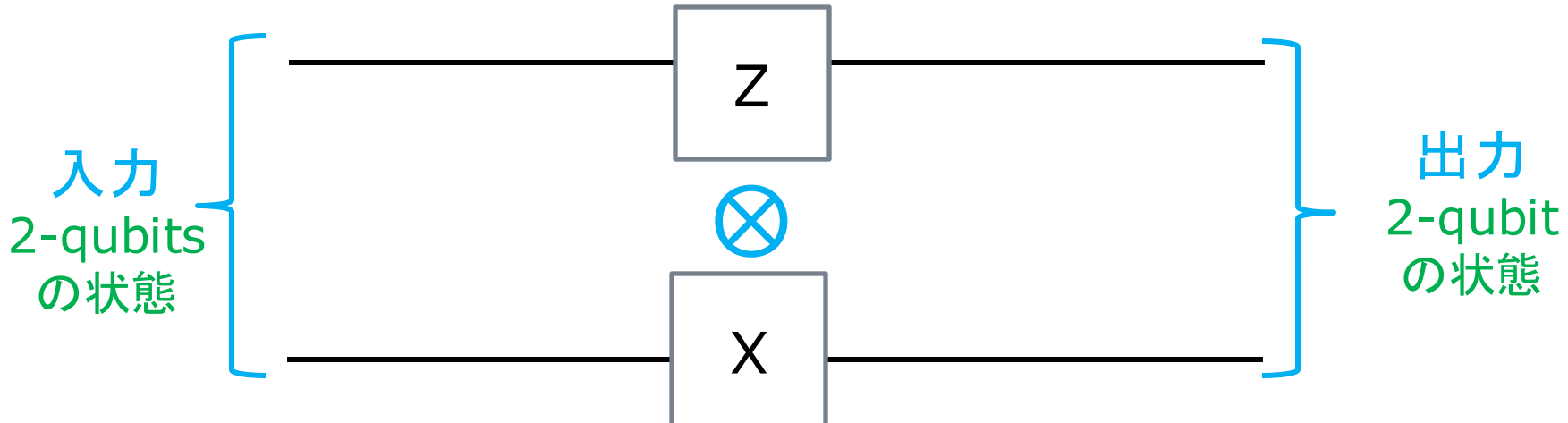


2-qubits ゲート ZX
入出力サンプル 2

- この結果は、2-qubitsゲートZXがなくて、二つの1-qubitゲートのパラレルに置かれたものの働きに等しい。




入力
2-qubits
の状態
Z
X
出力
2-qubit
の状態


入力
2-qubits
の状態
Z
X
出力
2-qubit
の状態

テンソル積の定義

行列のテンソル積

行列のテンソル積を次のように定義する。(2x2行列で例示)

$$A \otimes B = \begin{pmatrix} A_{11}B & A_{12}B \\ A_{21}B & A_{22}B \end{pmatrix}$$

Bの成分も書くと、

$$A \otimes B = \begin{pmatrix} A_{11}B_{11} & A_{11}B_{12} & A_{12}B_{11} & A_{12}B_{12} \\ A_{11}B_{21} & A_{11}B_{22} & A_{12}B_{21} & A_{12}B_{22} \\ A_{21}B_{11} & A_{21}B_{12} & A_{22}B_{11} & A_{22}B_{12} \\ A_{21}B_{21} & A_{21}B_{22} & A_{22}B_{21} & A_{22}B_{22} \end{pmatrix}$$

行列のテンソル積の例（1）

$A=\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix}$, $B=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}$ としよう。

$$A\otimes B=\begin{pmatrix}\mathbf{1} & \mathbf{-1}\\ \mathbf{0} & \mathbf{2}\end{pmatrix}\otimes\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}=\begin{pmatrix}\mathbf{1}\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix} & \mathbf{-1}\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}\\ \mathbf{0}\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix} & \mathbf{2}\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}\end{pmatrix}$$

$$=\begin{pmatrix}1 & 2 & -1 & -2\\ 3 & 4 & -3 & -4\\ 0 & 0 & 2 & 4\\ 0 & 0 & 6 & 8\end{pmatrix}$$

行列のテンソル積の例 (2)

$A=\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix}$, $B=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}$ としよう。

$$B\otimes A=\begin{pmatrix}\mathbf{1} & \mathbf{2}\\ \mathbf{3} & \mathbf{4}\end{pmatrix}\otimes\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix}=\begin{pmatrix}\mathbf{1}\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix} & \mathbf{2}\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix}\\ \mathbf{3}\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix} & \mathbf{4}\begin{pmatrix}1 & -1\\ 0 & 2\end{pmatrix}\end{pmatrix}$$

$$=\begin{pmatrix}1 & -1 & 2 & -2\\ 0 & 2 & 0 & 4\\ 3 & -3 & 4 & -4\\ 0 & 6 & 0 & 8\end{pmatrix}$$

テンソル積では、
$A\otimes B \neq B\otimes A$
である

ベクトルのテンソル積

ベクトルは、nx1の行列であるから、

$$\begin{pmatrix} \boldsymbol{a_1} \\ \boldsymbol{a_2} \end{pmatrix} \otimes \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{a_1} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \\ \boldsymbol{a_2} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} a_1 b_1 \\ a_1 b_2 \\ a_2 b_1 \\ a_2 b_2 \end{pmatrix}$$

ベクトルのテンソル積の例

$$\begin{pmatrix}\mathbf{1}\\\mathbf{2}\end{pmatrix}\otimes\begin{pmatrix}3\\4\end{pmatrix}=\begin{pmatrix}\mathbf{1}\begin{pmatrix}3\\4\end{pmatrix}\\\mathbf{2}\begin{pmatrix}3\\4\end{pmatrix}\end{pmatrix}=\begin{pmatrix}3\\4\\6\\8\end{pmatrix}$$

$$\begin{pmatrix}\mathbf{3}\\\mathbf{4}\end{pmatrix}\otimes\begin{pmatrix}1\\2\end{pmatrix}=\begin{pmatrix}\mathbf{3}\begin{pmatrix}1\\2\end{pmatrix}\\\mathbf{4}\begin{pmatrix}1\\2\end{pmatrix}\end{pmatrix}=\begin{pmatrix}3\\6\\4\\8\end{pmatrix}$$

2-qubitの基底

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$|10\rangle = |1\rangle \otimes |0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$|11\rangle = |1\rangle \otimes |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

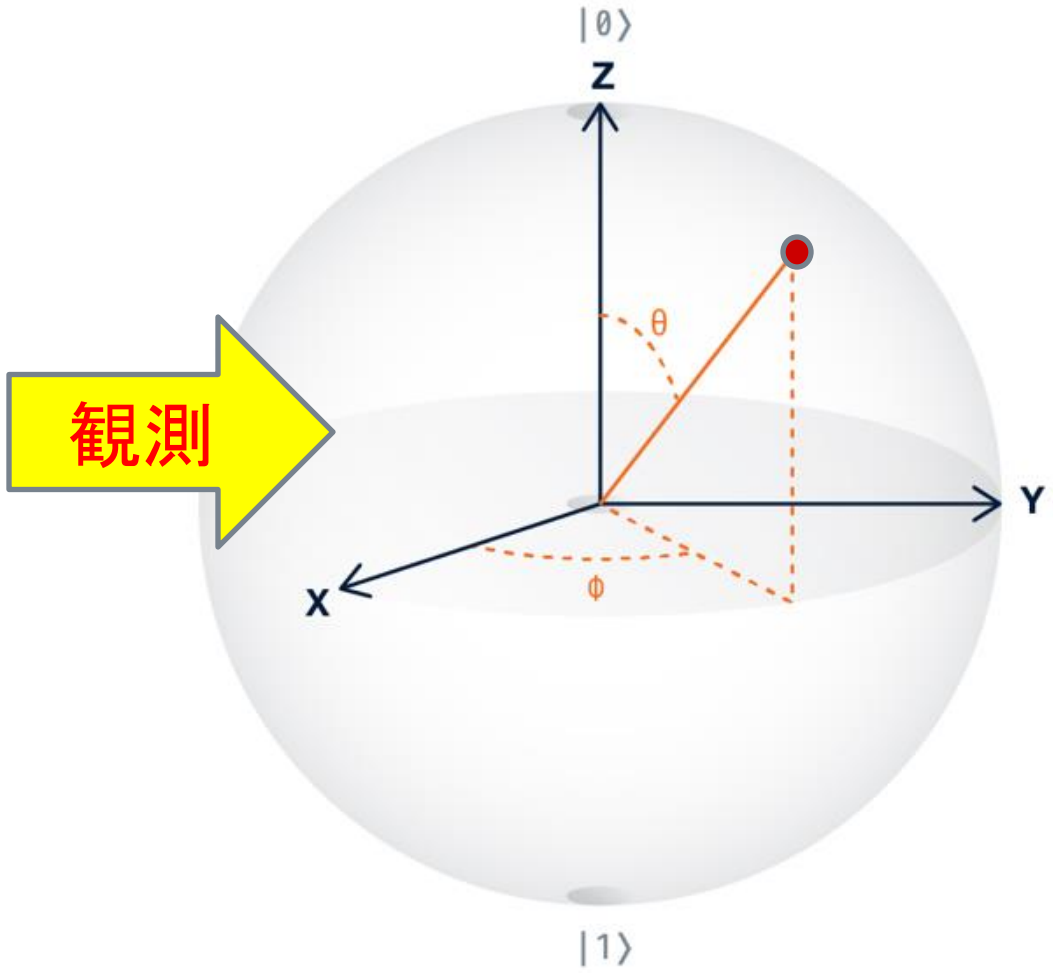
量子の不思議 1

– 量子の状態は観測できない？

量子の不思議 1
– 量子の状態は観測できない？

- qubitの観測
  - 観測による状態の変化
  - 観測の確率　Born ルール
- 一般の量子状態の観測
- 部分的な観測　2-qubitsの状態の観測
  - 部分的な観測　観測の確率
  - 部分的な観測　観測後の状態

Qubit

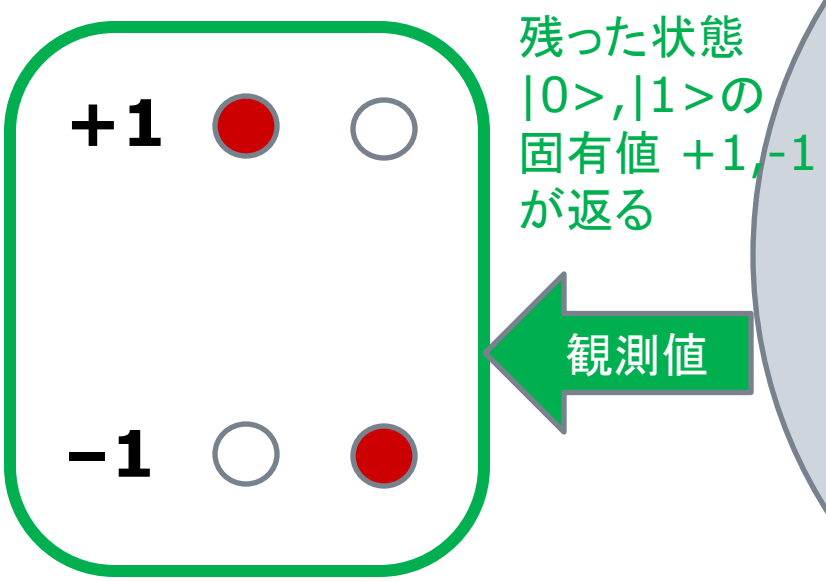


Qubitは、状態 |0>と状態 |1>の重ね合わせの状態を取る

$$|\mathrm{Qubit}\rangle = \alpha|0\rangle + \beta|1\rangle$$

$$|\alpha|^2 + |\beta|^2 = 1\ ;\quad \alpha, \beta \in \mathbb{C}$$

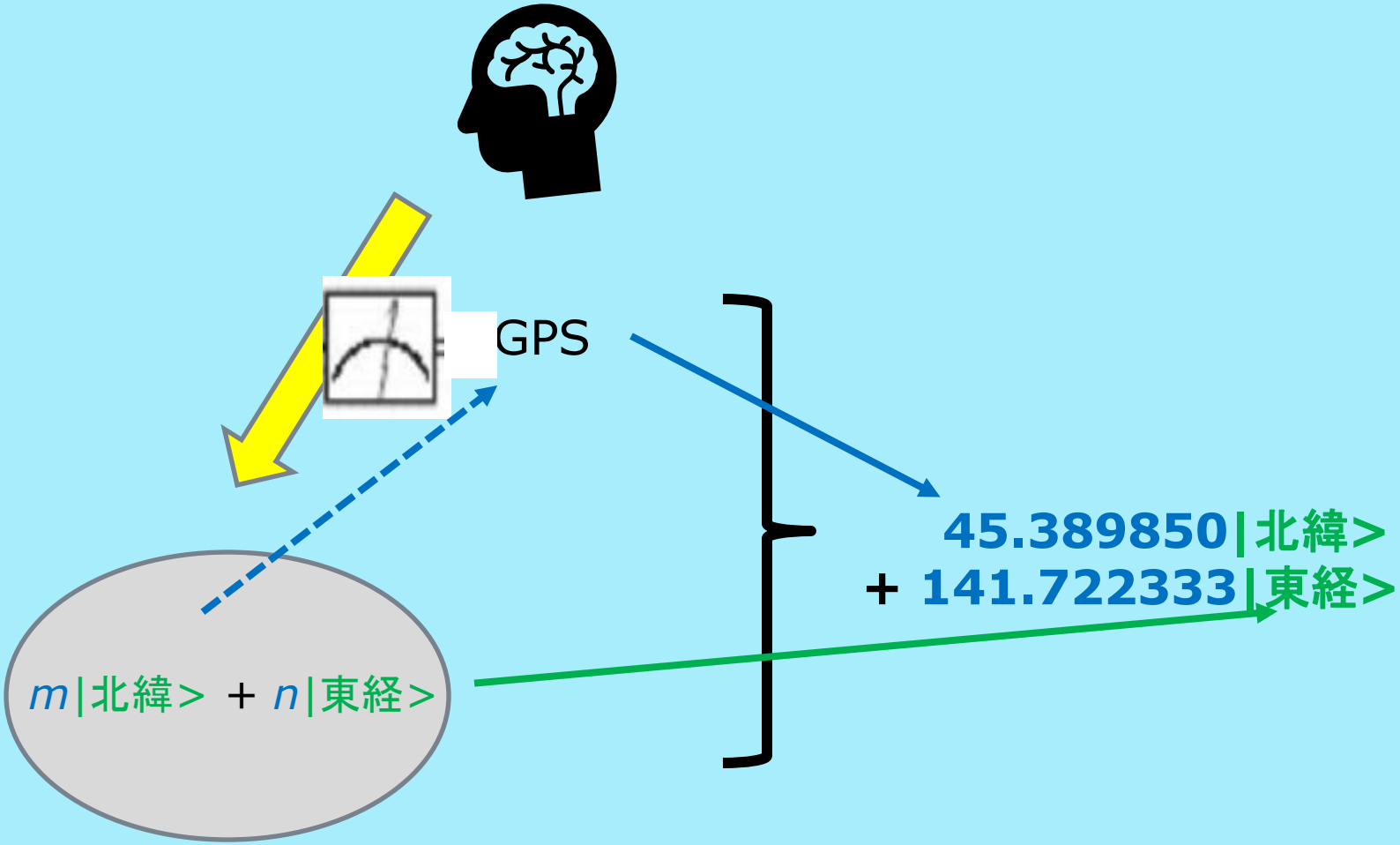
観測を行うと、Qubitの
重ね合わせの状態は
失われ、|0>か|1>かの
状態が残る。



この時
|0>が残る確率は、$|\alpha|^2$で、
|1>が残る確率は、$|\beta|^2$で、
与えられる。

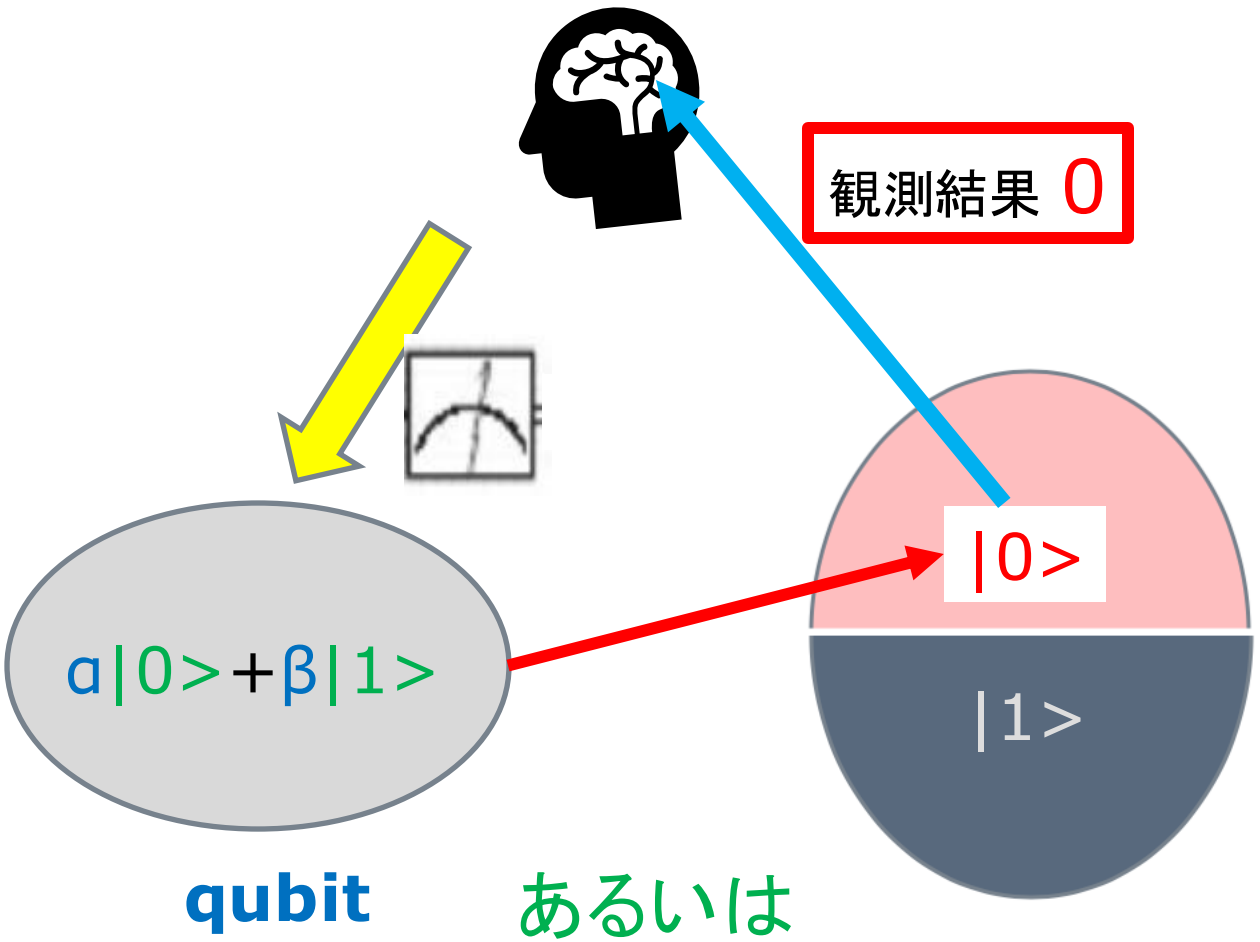
$$\alpha^*\alpha + \beta^*\beta = 1 \; ; \quad \alpha, \beta \in \mathbb{C}$$

比較：GPSで位置情報を観測する



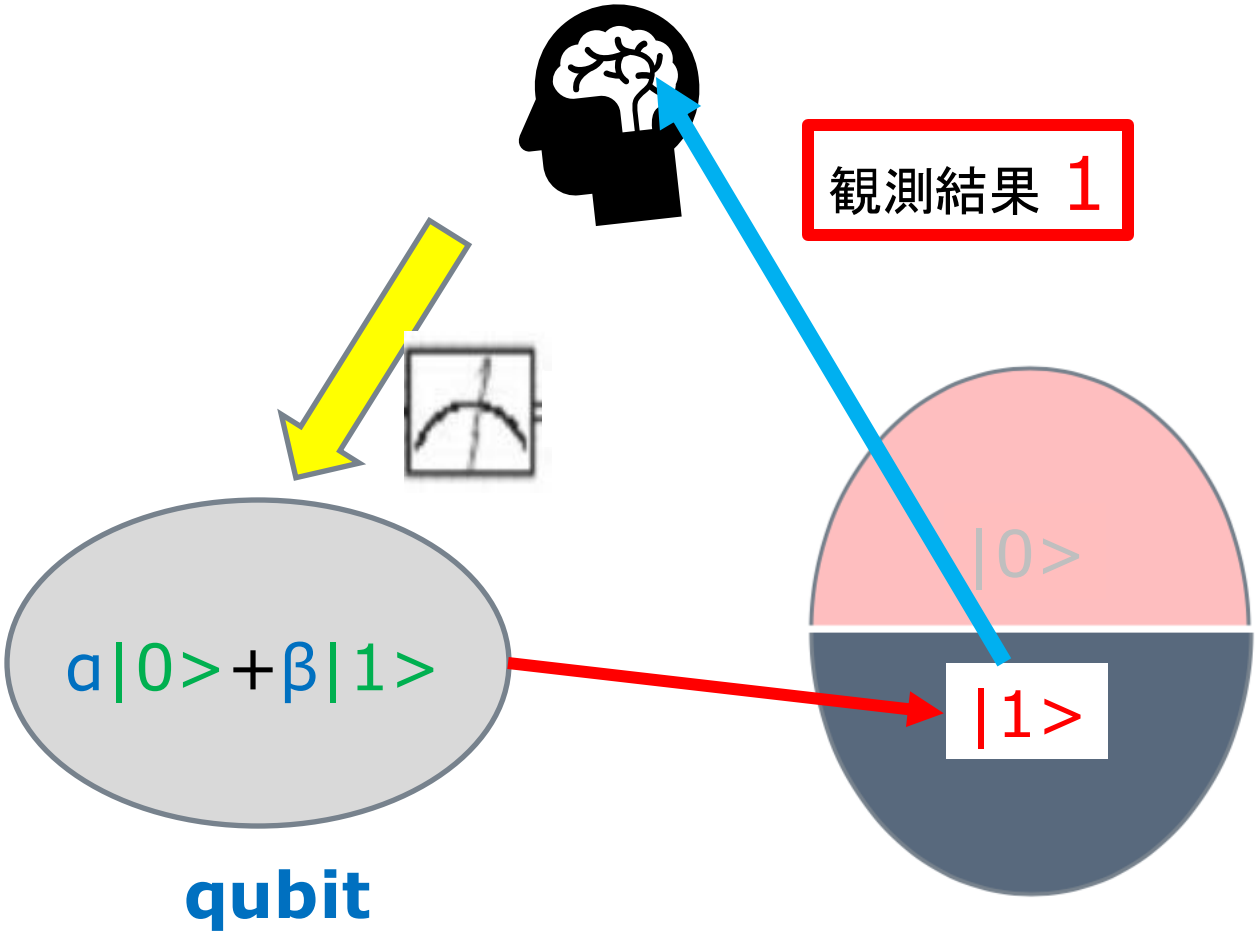
成分 m, n は、観測できている

qubitを観測する



成分 α, β は、観測にかからない

qubitを観測する



成分 α, β は、観測にかからない

qubitの状態の観測

- qubitの状態は、一つの数字ではなく二つの数字で表される
- qubitの状態は、0の状態 |0>と1の状態 |1>の「重ね合わせ」である。

- qubitを「観測」すると、「重ね合わせ」の状態は失われて、|0>または|1>いずれかの状態が残る。残された二つの状態の観測値は、2値のビットのように見える。

- 一回の観測では、0か1しか返ってこないのだが、繰り返し観測すれば、0が観測される確率と1が観測される確率は一定の値に近づいていく。これが、元の量子ビットの|0>の状態と|1>の状態の「重ね合わせ」の情報を与えると考えることができる。

観測と観測による状態の変化
観測の確率　Born ルール

- 量子の状態は、直接には、観測できない。
- また、観測によって、系の状態は、観測前とは異なる状態に変化する。（以前の重ね合わせは失われる。）
  - α|0> ＋ β|1>の重ね合わせの状態は、観測によって|0>または|1>の新しい状態に変化する。（「崩壊する」）
- 新しい状態が観測される確率は、正確に計算することができる。Bornのルール
  - |0>が観測される確率は、$|\alpha|^2$ で、
    |1>が観測される確率は、$|\beta|^2$ で、与えられる。

一般の量子状態の観測

一般の量子状態 $|\psi\rangle = \sum_{k=0}^{n-1} c_k|k\rangle$ が与えられた時、

状態 $|0\rangle$ が観測される確率は、$|c_0|^2$ で与えられる。
観測前の状態 $|\psi\rangle$は、新しい状態 $|0\rangle$に変わる。

状態 $|1\rangle$ が観測される確率は、$|c_1|^2$ で与えられる。
観測前の状態$|\psi\rangle$は、新しい状態 $|1\rangle$に変わる。

…

状態 $|k\rangle$ が観測される確率は、$|c_k|^2$ で与えられる。
観測前の状態$|\psi\rangle$は、新しい状態 $|k\rangle$に変わる。

部分的な観測
2-qubitsの状態の観測

2-qubits　部分的な観測
観測の確率

- $|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ で、最初の1ビットのみを観測して、それが0である確率は、どうなるであろうか？

- 次のように考える。
  最初の 1 ビットが 0である状態は、|00>と|01>である。
  最初の1ビットのみを観測して、それが0である確率は、|00>を観測した場合か、|01>を観測した場合のいずれかである。
  よってその確率は、それらを加えたものになる。

$$\Pr\{1\text{st bit} = 0\} = \Pr\{00\} + \Pr\{01\} = |\alpha_{00}|^2 + |\alpha_{01}|^2.$$

2-qubits　部分的な観測
観測後の状態

- 最初の1ビットのみを観測して、それが0であったとする。このとき、2-Qubitsの状態は、観測後どういう状態になるのだろうか？
- それは、この観測に矛盾する全ての項（最初のビットが1の項）を消して、残った項の重ね合わせの状態になる。

- 最初の1ビットが0である項 |00>, |01>を残し、それに矛盾する最初の1ビットが1である項 |10>, |11>を消すと、次のようになる。

観測前の状態：$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$

観測後の状態：$|\psi\rangle' = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$

2-qubits　部分的な観測
観測後の状態

- ただ、それが、単位ベクトルになるように、正規化されねばならない。新しい状態は、次のようになる。

$$|\phi_{\text{new}}\rangle = \frac{\alpha_{00}|00\rangle + \alpha_{01}|01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}$$

- これは、ちょっと複雑に思えるが、本質的には、この状態は、$\alpha_{00}|00\rangle + \alpha_{01}|01\rangle$ で、それに正規化の係数

$$\frac{1}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}$$

がかかったものと考えればいい。

2-qubits　部分的な観測
観測後の状態

元の状態を $|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ として、正規化因子を無視すると、観測後の状態は、次のようになる。

1. 第1ビットのみが0と観測された場合

観測後の状態：$|\psi\rangle' = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$

2. 第1ビットのみが1と観測された場合

観測後の状態：$|\psi\rangle' = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$

3. 第2ビットのみが0と観測された場合

観測後の状態：$|\psi\rangle' = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$

4. 第2ビットのみが1と観測された場合

観測後の状態：$|\psi\rangle' = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$

$|GHZ\rangle = 1/\sqrt{2}\ (|000\rangle + |111\rangle)$の場合

1. 第一qubitが、0と観測された時、観測後に状態はどのように変わるか？
2. この時、第二qubitが1と観測される確率を求めよ。
3. この時、第三qubitが1と観測される確率を求めよ。
4. 第一qubitが、1と観測された時、観測後に状態はどのように変わるか？
5. この時、第二qubitが1と観測される確率を求めよ。
6. この時、第三qubitが1と観測される確率を求めよ。

量子の不思議 2

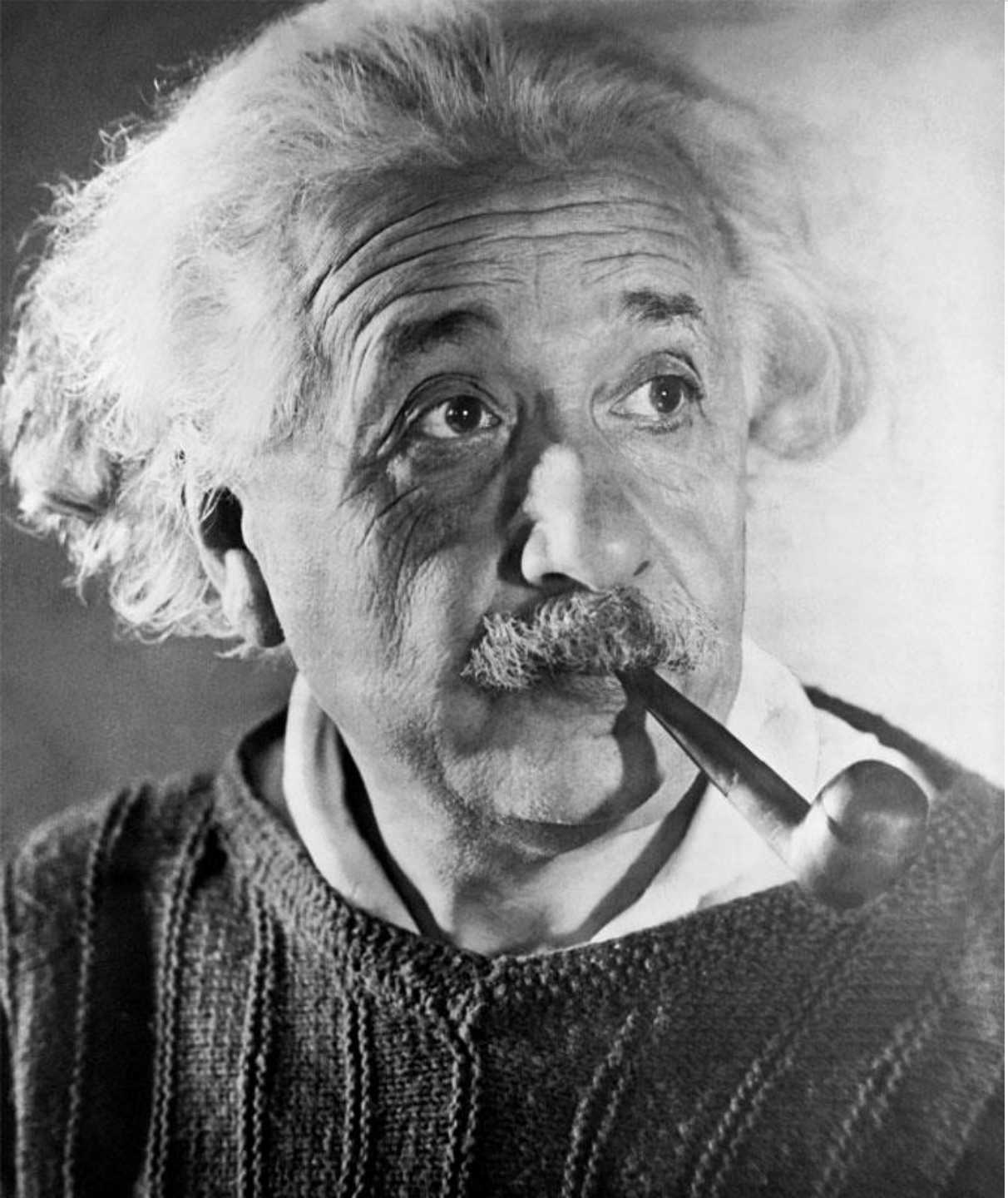
-- エンタングルメント --

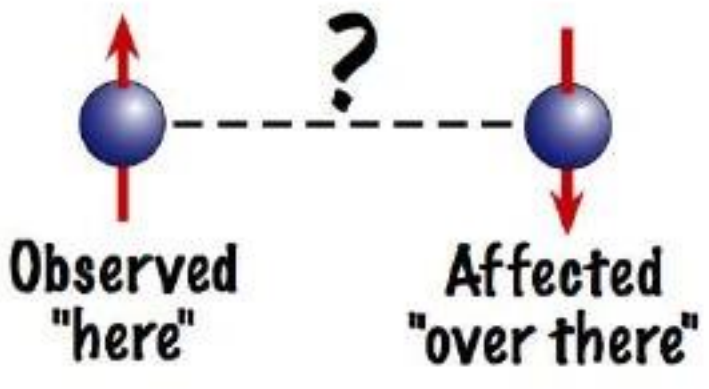
参考資料
「エンタングルメントで理解する量子の世界」
https://www.marulabo.net/docs/entangle-talk/

量子の不思議 2
-- エンタングルメント --

- アインシュタインの発見
- エンタングルメント
  二つの1-qubitのテンソル積に分解できない状態
- EPRペアの観測
- 観測後の状態
- 「馬鹿げた遠隔作用」？

アインシュタインの発見





1935年

EPRの逆理

Einstein,
Podolsky,
Rosen

エンタングルメント
もつれあった二つの量子の状態の発見

- 1935年に、アインシュタインとポドルスキーとローゼンは、次の論文を発表する。（三人の著者の頭文字をとって、EPR論文と呼ばれる。）
- "Can Quantum-Mechanical Description of Physical Reality Be Considered Complete ?"「物理的な実在の量子力学の記述は、完全なものと考えることができるか？」 https://goo.gl/qAWacP
- この論文で、アインシュタインは、量子論では、二つの量子の「もつれあい」の状態が現れることを指摘した。

EPR論文を報ずる1935年のニューヨークタイムズ紙

EINSTEIN ATTACKS QUANTUM THEORY

Scientist and Two Colleagues Find It Is Not 'Complete' Even Though 'Correct.'

SEE FULLER ONE POSSIBLE

Believe a Whole Description of 'the Physical Reality' Can Be Provided Eventually.

エンタングルメント

二つの1-qubitのテンソル積に分解できない状態

二つの1-qubitのテンソル積に分解できない状態
エンタングルメント

- すべての2-qubitsの状態が、二つの1-qubitのテンソル積に分解できるとは限らない。二つの1-qubitのテンソル積に分解できない2-qubitの状態を、**エンタングルメント** という。
  そういう状態があることを、次に見ていこう。

- ある2-qubitの状態が二つの1-qubitのテンソル積に分解できるか否かは、二つの1-qubitのテンソル積で表現される2-qubitの状態が、次の係数を持つこと利用してチェックできる。

$$ac|00\rangle+ad|01\rangle+bc|10\rangle+bd|11\rangle$$

この係数を、2-qubitの状態の係数と比較すればいい。

$1/\sqrt{2}$ (|00>+|11>)の場合

- $1/\sqrt{2}$ (|00>+|11>)
  =$1/\sqrt{2}$(|00> □ □ + |11>)
  =? ac|00>+ad|01>+bc|10>+bd|11>

  |01>, |10>の項が含まれていないので、ad=bc=0。
  これから aとd、bとcのいずれかが0であることがわかる。
  この時、ac あるいは bd のいずれかは0になるので、二つの係数は一致しない。

- $1/\sqrt{2}$ (|00>+|11>) はエンタングルメント状態である。

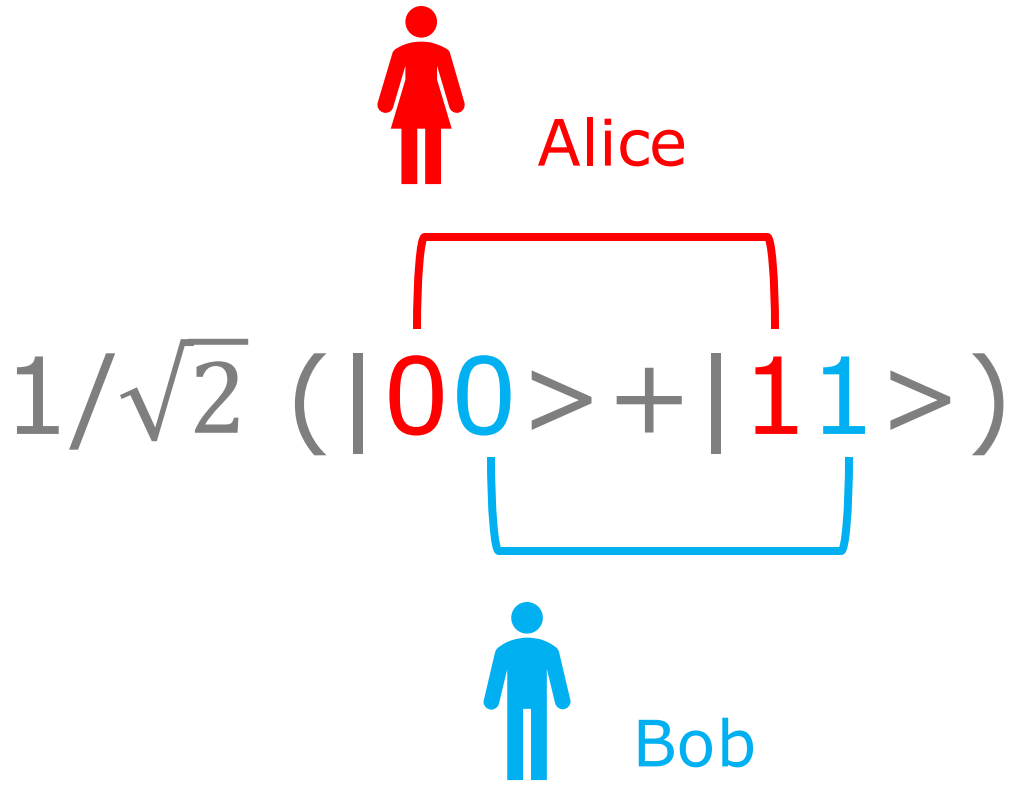
EPRペア
二つのqubitの状態が一つの式で表される

- アインシュタインらが発見した、もつれあった二つの量子の状態を、発見者の頭文字をとって「EPRペア」という。それは、次の4種類ある。これを、Φ^+, Φ^-, Ψ^+, Ψ^- と呼ぶことがある。

  - Φ^+ : $1/\sqrt{2}\ (|00\rangle+|11\rangle)$
  - Φ^- : $1/\sqrt{2}\ (|00\rangle-|11\rangle)$
  - Ψ^+ : $1/\sqrt{2}\ (|01\rangle+|10\rangle)$
  - Ψ^- : $1/\sqrt{2}\ (|01\rangle-|10\rangle)$

- 通常は、二つのqubitの状態は、それぞれのqubitの状態を表す二つの式で表されるのだが、EPRペアの場合、二つのqubitの状態が一つの式で表されている。

EPRペア：もつれ合った二つのqubit

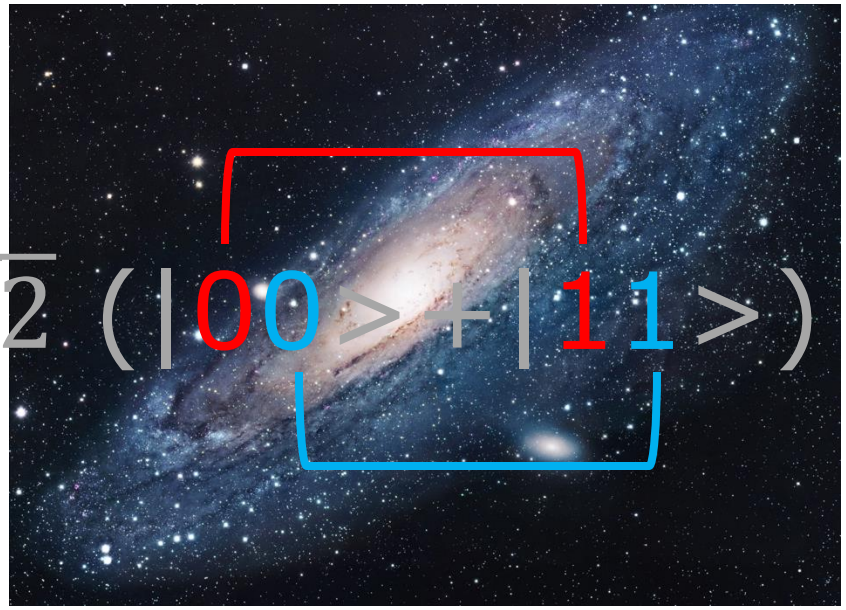


$1/\sqrt{2}$ (|00>+|11>) で表される状態は、二つのqubitの状態である。
一方のqubitをAliceが、他方のqubitをBobが持つことができる。

EPRペア：もつれ合った二つのqubit

Alice

$1/\sqrt{2}\ (|00>+|11>)$



Bob

$1/\sqrt{2}\ (|00>+|11>)$

Aliceが観察できるのは、$|00>+|11>$の第一bitで、
Bobが観察できるのは、$|00>+|11>$の第二bitである。
この関係は、両者がどんなに離れていても変わりない。

EPRペアの観測

EPRペア $1/\sqrt{2}\ (|00>+|11>)$ を観測する

AliceとBobがEPRペア $1/\sqrt{2}\ (|00>+|11>)$ で表される状態のqubitを持っているとしよう。Aliceが持っているqubitは、EPRペアの片割れで、その状態は、次の赤字で示す状態に対応している。　$1/\sqrt{2}\ (|00>+|11>)$

AliceがEPRペア $1/\sqrt{2}\ (|00>+|11>)$を観測したとしよう。Aliceが観測できるのは、自分が持つqubitの状態だけである。

それが|0>である確率は、$1/\sqrt{2}(|00>+|11>)$から、|01>の係数 $1/\sqrt{2}$ の絶対値の二乗で1/2となる。同様に、

それが|1>である確率は、$1/\sqrt{2}(|00>+|11>)$から、|11>の係数 $1/\sqrt{2}$ の絶対値の二乗で1/2となる。

EPRペア $1/\sqrt{2}\,(|00\rangle+|11\rangle)$ を観測する 観測後の状態

Aliceが、EPRペア $1/\sqrt{2}\,(|00\rangle+|11\rangle)$ を観測して、その結果が$|0\rangle$であったとしよう。

$$1/\sqrt{2}\,(|00\rangle+\cancel{|11\rangle})$$

この観測の結果、新しい状態は、$|00\rangle$に変わる。

それは、第二ビットの観測が、$|0\rangle$である確率が 1 であることを意味する。100% の確率で、Bobの持つ第二ビットの状態が0であることがわかる。

すなわち、Aliceが自分のqubitで状態$|0\rangle$を観測するとすぐに、遠く離れたBobの持つqubitの状態が$|0\rangle$であることがわかることになる。

EPRペア $1/\sqrt{2}\,(|00\rangle+|11\rangle)$ を観測する 観測後の状態

Aliceが、EPRペア $1/\sqrt{2}\,(|00\rangle+|11\rangle)$ を観測して、その結果が$|1\rangle$であったとしよう。

$1/\sqrt{2}\,(\not{|00\rangle}+|11\rangle)$

この観測の結果、新しい状態は、$|11\rangle$に変わる。

それは、第二ビットの観測が、$|1\rangle$である確率が 1 であることを意味する。100% の確率で、Bobの持つ第二ビットの状態が$|1\rangle$であることがわかる。

すなわち、Aliceが自分のqubitで状態$|1\rangle$を観測するとすぐに、遠く離れたBobの持つqubitの状態が$|1\rangle$であることがわかることになる。

「馬鹿げた遠隔作用」?

Aliceの観測結果をまとめると、次のようになる。

Aliceが自分のqubitで状態|0>を観測すると、すぐに、遠く離れたBobの持つqubitの状態が|0>であることがわかる。

Aliceが自分のqubitで状態|1>を観測すると、すぐに、遠く離れたBobの持つqubitの状態が|1>であることがわかることになる。

Aliceの観測結果が、瞬時に、遠く離れたBobの観測結果に影響を与える? これは、光のスピード以上で情報が伝わらないとする物理法則に矛盾しないか?

実際、アインシュタインは、こうした現象は「馬鹿げた遠隔作用」だと言った。

Part II

アダマールゲートで学ぶ
量子アルゴリズムの基礎

Part II Agenda

アダマールゲートで学ぶ量子アルゴリズムの基礎

1. 量子コインと量子暗号
2. Quantum Parallelism
3. エンタングルメントと量子通信

量子コインと量子暗号

参考資料
「暗号技術の現在 ー ポスト量子暗号への移行と量子暗号」
https://www.marulabo.net/docs/cipher/

量子コインと量子暗号

- 量子コイン　H|0>, H|1>
- 計算基底とアダマール基底
- 量子暗号の基礎 – 秘密鍵の共有

量子コイン　H|0>

|0>をアダマール行列Hに作用させた結果を観測してみよう。

$H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ である。

|0> が観測される確率は、$|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$

|1> が観測される確率は、$|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$

だから、|0>と|1>は、同じ確率 1/2 で観測される。

これを「量子コイン」と呼ぶ。

量子コイン H|1>

|1>をアダマール行列Hに作用させた結果を観測してみよう。

$H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$ である。

|0> が観測される確率は、$|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$

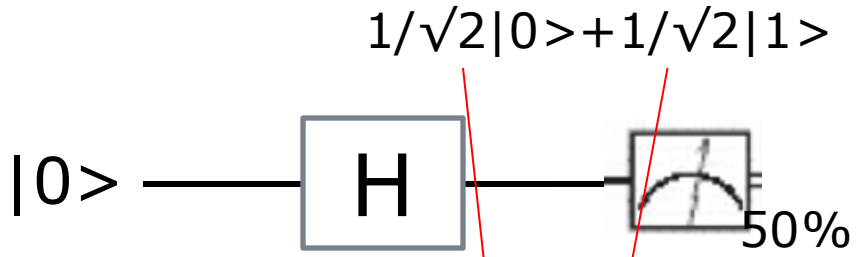
|1> が観測される確率は、$|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$

だから、|0>と|1>1は、同じ確率で観測される。

H|0>だけでなく、H|1>も「量子コイン」である。

このことは、また、観測によっては、H|0>とH|1>を、区別できないことを意味する。

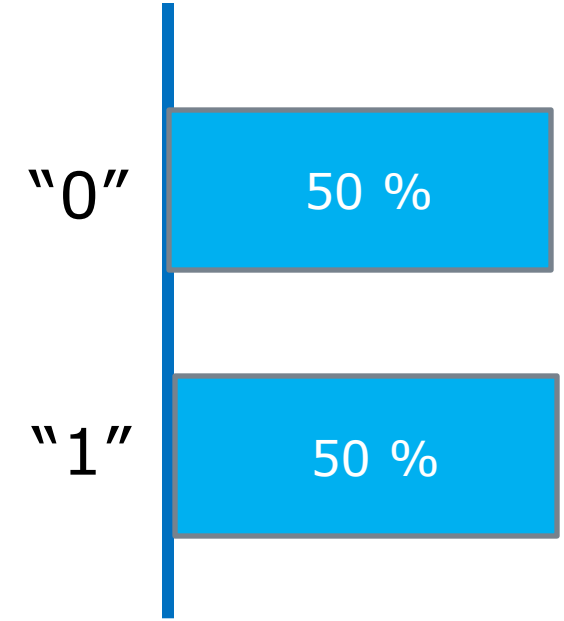
つぎのように、アダマール・ゲートH 一つに、
初期値として |0>を与えた時、サンプリングで得られる分布を考えよう



Hは、|0>を
1/√2|0>+1/√2|1>
に変える

この時、|0>が観測される確率は
$|1/\sqrt{2}|^2 = 1/2$
この時、|1>が観測される確率は
$|1/\sqrt{2}|^2 = 1/2$

よって、分布は次のようになる



計算基底 |0>, |1>と アダマール基底 |+>, |−>

$|+\rangle = H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ 、

$|-\rangle = H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$ とする。

この時 |+>, |−>は、正規直交基底になる。
これをアダマール基底と呼ぶ。
（<+|+>=<−|−>=1, <+|−>=<−|+>=0 を確かめよ ）
また、

H|+> = |0>、H|−> = |1> が成り立つ。

$$H|+\rangle = H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{\sqrt{2}}(H|0\rangle + H|1\rangle)$$
$$= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle + \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) = |0\rangle$$
$$H|-\rangle = H\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{\sqrt{2}}(H|0\rangle - H|1\rangle)$$
$$= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle - \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) = |1\rangle$$

量子暗号の基礎

Alice: 古典bitをqubitにエンコードして qubitをBobに送る

- Aliceには、古典bitをqubitにエンコードするのに、次の二つの方法を利用できる。

1. |0>, |1>を基底としてエンコードする。
   この時、古典bit 0 は、qubit |0>に、
   古典bit 1 は、qubit |1>にエンコードされる。
2. |+>, |−>を基底としてエンコードする。
   この時、古典bit 0 は、qubit |+>に、
   古典bit 1 は、qubit |−>にエンコードされる。

- 基底 |0>, |1> と基底 |+>, |−>の関係については、あとで詳しく述べる。

Bob: Aliceからqubitを受け取り、観測して古典bitにデコードする

- Aliceには、qubitを古典bitにデコードするのに、次の二つの方法を利用できる。

1. |0>, |1>を基底としてqubitを観測し、デコードする。
   この時、qubit |0> は、古典bit 0 に、
   qubit |1> は、古典bit 1 にデコードされる。
2. |+>, |−>を基底としてqubitを観測し、デコードする。
   この時、qubit |+> は、古典bit 1に、
   qubit |−> は、古典bit 0にエンコードされる。

基底 |0>, |1> と基底 |+>, |−>の関係

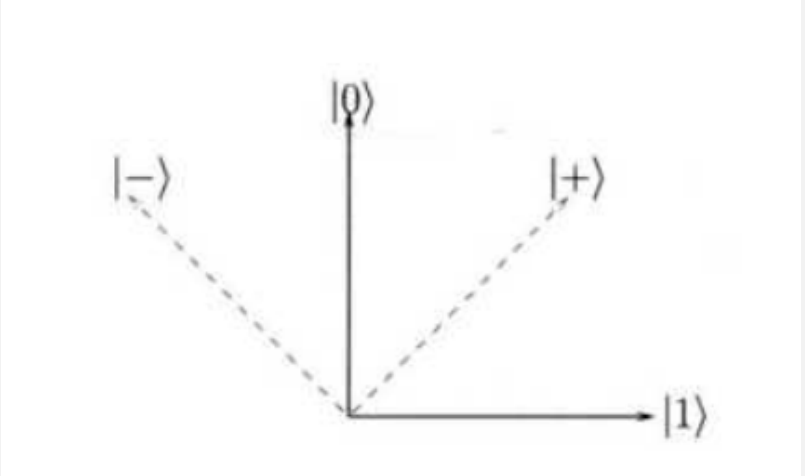
- 基底 |0>, |1> と基底 |+>, |−>の間には、つぎのような関係がある。

1. $|0\rangle = \frac{1}{\sqrt{2}}|+\rangle + \frac{1}{\sqrt{2}}|-\rangle$

   $|1\rangle = \frac{1}{\sqrt{2}}|+\rangle - \frac{1}{\sqrt{2}}|-\rangle$

2. $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$

   $|-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$



基底 |0>, |1>を「標準基底」、
基底 |+>, |−>を「アダマール基底」と呼ぶ。

Aliceがエンコードに選んだ基底と Bobがデコードに選んだ基底が同じ場合

- AliceとBobが共に「標準基底」を選んだ場合
  - 古典bitが0の場合、AliceはBobにqubit |0>を送る。
    |0> =1・|0>+0・|1>だから、Bobは100%の確率で|0>を観測するので、0にデコードできる。
  - 古典bitが1の場合、AliceはBobにqubit |1>を送る。
    |1> =0・|0>+1・|1>だから、Bobは100%の確率で|1>を観測するので、1にデコードできる。
- AliceとBobが共に「アダマール基底」を選んだ場合
  - 古典bitが0の場合、AliceはBobにqubit |+>を送る。
    |+> =1・|+>+0・|->だから、Bobは100%の確率で|+>を観測するので、0にデコードできる。
  - 古典bitが1の場合、AliceはBobにqubit |->を送る。
    |-> =0・|+>+1・ |->だから、Bobは100%の確率で|->を観測するので、1にデコードできる。
- 同じ基底を選んだ時には、正しく情報を伝えられる。

Aliceがエンコードに選んだ基底と Bobがデコードに選んだ基底が異なる場合

- Aliceが「標準基底」、Bobが「アダマール基底」を選んだ場合
  - 古典bitが0の場合、AliceはBobにqubit |0>を送る。
    $|0> = \frac{1}{\sqrt{2}}|+> + \frac{1}{\sqrt{2}}|->$だから、Bobは 1/2の確率で|+>を、同じく 1/2 の確率で|−>を観測するので、0か1か決まらない。
  - 古典bitが1の場合、AliceはBobにqubit |1>を送る。
    $|1> = \frac{1}{\sqrt{2}}|+> - \frac{1}{\sqrt{2}}|->$だから、1/2の確率で|+>を、同じく 1/2 の確率で|−>を観測するので、0か1か決まらない。
- Aliceが「アダマール基底」、Bobが「標準基底」を選んだ場合
  - 古典bitが0の場合、AliceはBobにqubit |+>を送る。
    $|+> = |+> = \frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$だから、Bobは 1/2の確率で|0>を、同じく 1/2 の確率で|1>を観測するので、0か1か決まらない。
  - 古典bitが1の場合も同様である。
- 異なる基底を選んだ時には、正しく情報が伝えられない。

AliceとBobで、秘密鍵を共有する

- Aliceは、ランダムに 0, 1のビット列を作っておく。
- Aliceは、ランダムに「標準基底」「アダマール基底」を選んで、生成したビット列を1ビットごとにqubitにエンコードしてBobに送る。
- Bobは、ランダムに「標準基底」「アダマール基底」を選んで、送られてきたqubitをビットにデコードする。
- 作業が終わったら、AliceとBobは、それぞれどの基底をエンコード、デコードに使ったかを情報交換する。
- AliceとBobが、エンコードとデコードで同じ基底を使ったビットのみを抜き出して、それを秘密鍵とする。
- AliceとBobが、どのビットで同じ基底を使ったという情報は、第三者に漏れても構わない。同じ基底を使ったというだけでは、そのビットが0だったのか1だったかはわからないからである。

エンコード、デコードの例

Aliceが最初に用意したビット列

Bit	x_1	x_2	x_3	x_4
Value	0	1	1	0

＋：標準基底
Ｘ：アダマール基底

Aliceが選んだ基底とエンコードされたqubit

Classical value	0	1	1	0
Alice's basis	+	×	+	×
Quantum encoding	$\|\psi_1\rangle = \|0\rangle$	$\|\psi_2\rangle = \|-\rangle$	$\|\psi_3\rangle = \|1\rangle$	$\|\psi_4\rangle = \|+\rangle$

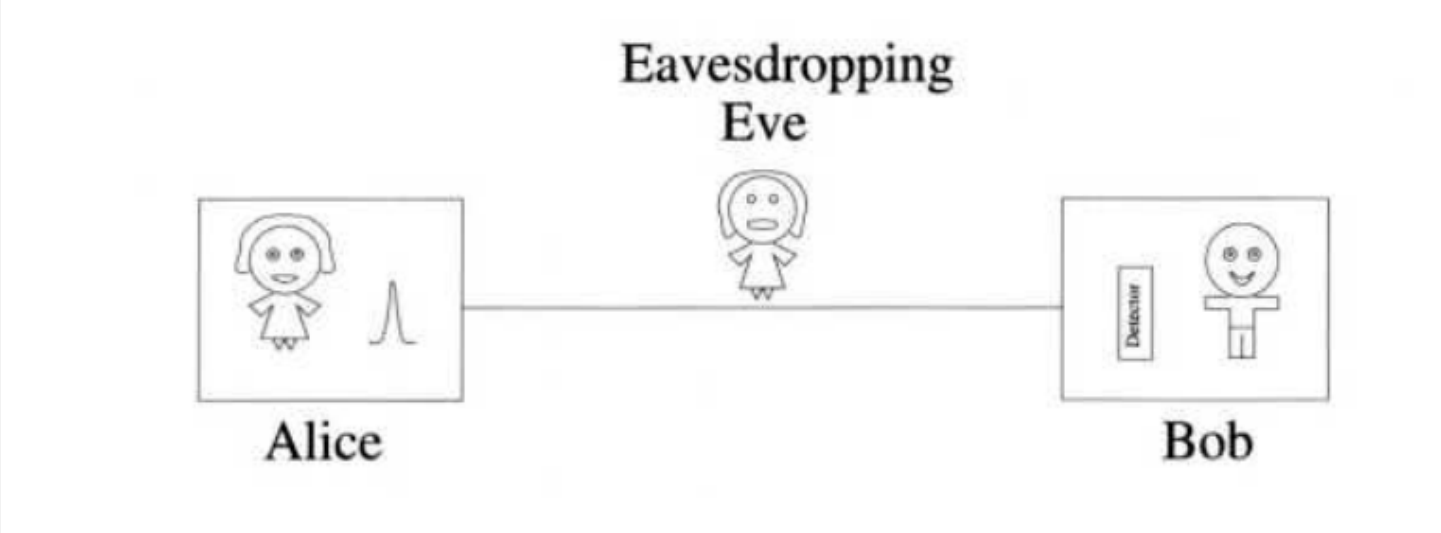
Aliceが選んだ基底とBobの選んだ基底は一致しているか？

Classical value	0	1	1	0
Alice's basis	+	×	+	×
Bob's basis	×	×	+	+
In agreement	No	Yes	Yes	No

両者の基底が一致したビットだけ、この例では、'1', `1' を、両者で共有する。

中間に盗聴者Eveがいた場合

この方式は、AliceとBobの通信の中間に盗聴者がいたとしても、AliceとBobは、そのことに確率的に気づくことができる。以下、それを見よう。



中間に盗聴者Eveがいた場合

- Eveが、AliceとBobが共有する秘密キーの情報を獲得する確率を考える。

1. 共有されるビットは、AliceとBobが同じ基底を使ったビットである。ただ、Eveは、その基底を知らないので、あてずっぽうに二つの基底から一つを選ぶしかない。この時点で、Eveは50%は推測に成功するが、50%は失敗する。
2. Eveは、基底を知らないだけでなく、Aliceが送ったビットが、0なのか1なのかを知らない。ここでも、彼は50%は失敗する。
3. 結局、EveがAliceとBobの基底を正しく推測し、Aliceが送ったビットを正しくBobに送るのに成功する確率は、0.5 x 0.5 =25%にとどまる。

- AliceとBobが、いくつかの秘密キーのビットをチェックすれば、高い確率で、盗聴が行われたことを検知できる。

Quantum Parallelism

参考資料
「3時間で学ぶShorのアルゴリズム入門」
https://www.marulabo.net/docs/shor/

Quantum Parallelism

- n個のqubitの状態の数
- アダマールゲート H を並列に配置する
- 任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形
- Quantum Parallelism
- $\Sigma|x> f(|x>)$の観測
- 何かが必要？
- ランダム量子回路

n個のqubitの状態の数

一個のqubitの状態

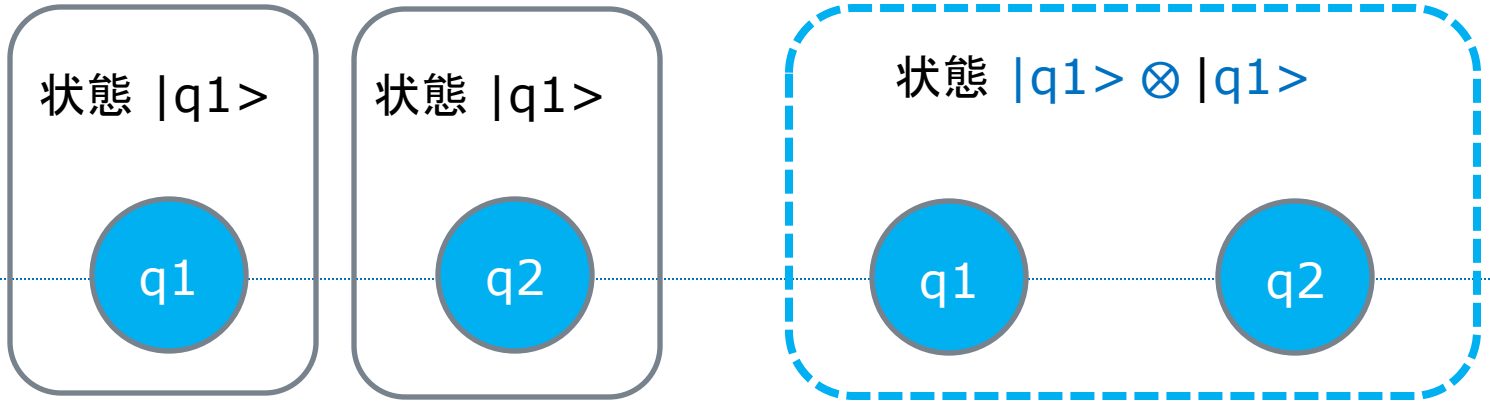
- 一個のqubitは、0の状態と1の状態の「重ね合わせ」の状態を持つ。二つの状態が重なり合っているが、一つの状態である。
- 0の状態を |0>
  1の状態を |1> で表す。
- qubitの重ね合わせの状態で、
  0らしさを表す数を a
  1らしさを表す数を b としたとき、
  このqubit q1の状態を次のように表すことにする。

$$|q1\rangle = a|0\rangle + b|1\rangle$$

- 全てのqubitの状態は、二つの数 a, b と |0>, |1> を使って、先の形で表される。
- ただし、a, b には $|a|^2+|b|^2 = 1$ という条件がつく。

二個のqubitの状態

- 二つのqubit q1とq2があったとする。
- q1とq2の状態は、それぞれ、次のように表される。
  |q1> = a|0> + b|1>
  |q2> = c|0> + d|1>
- この時、二つのqubit q1とq2を、一緒に考えた状態を考えて、それを、状態 $|q1\rangle \otimes |q2\rangle$ と表すことにする。

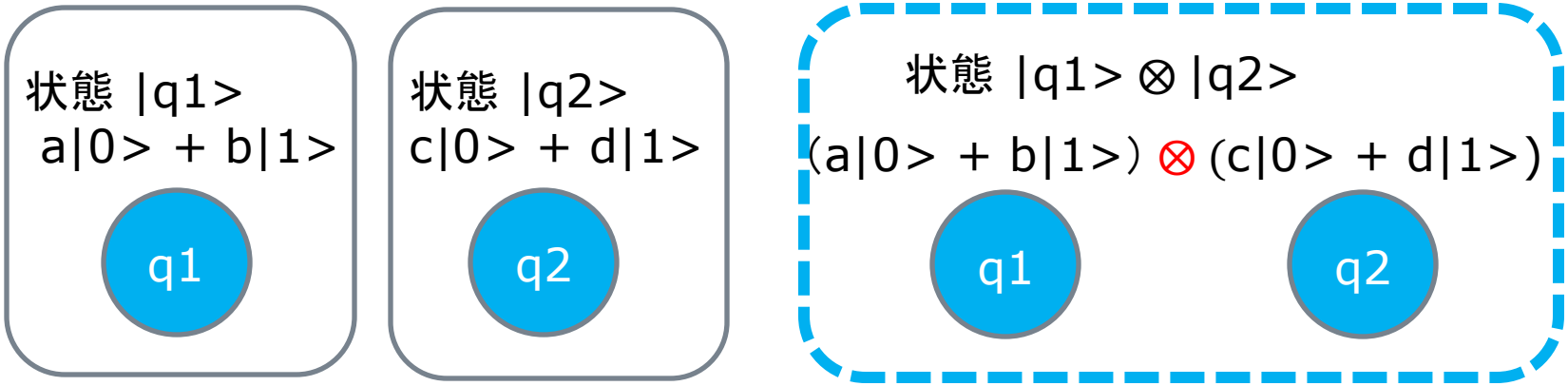


二個のqubitの状態の計算

- この時、|q1> ⊗ |q2> を、次のように計算する。

$$|q1\rangle \otimes |q2\rangle = (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle)$$

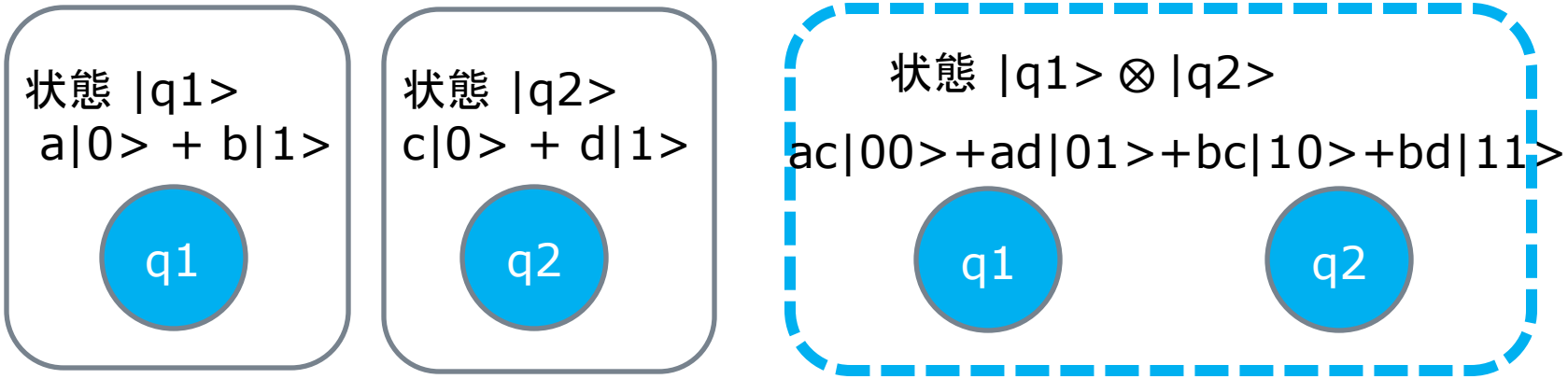
$$= ac|0\rangle\otimes|0\rangle + ad|0\rangle\otimes|1\rangle + bc|1\rangle\otimes|0\rangle + bd|1\rangle\otimes|1\rangle$$



二個のqubitの状態の数

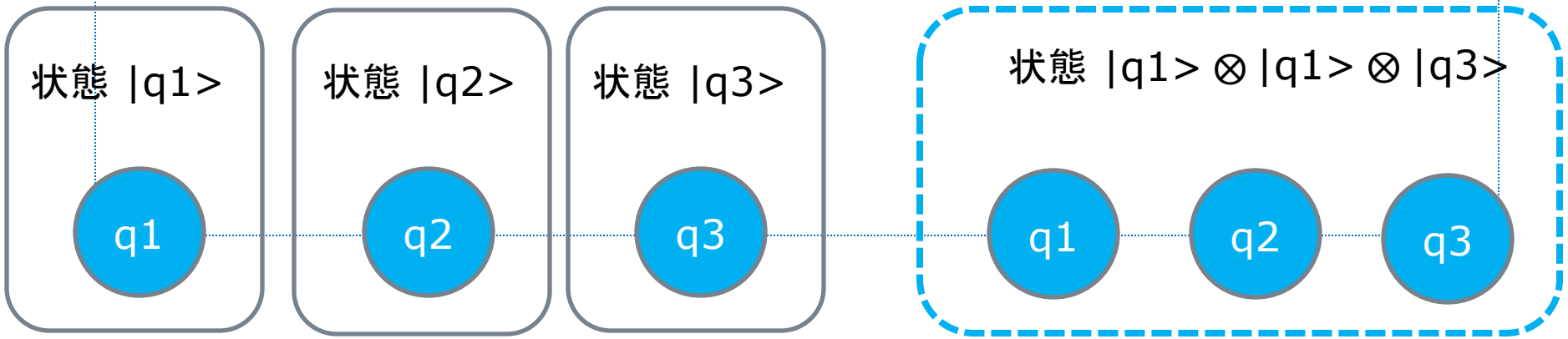
- 先の式で、|0>⊗|0>=|00>, |0>⊗|1>=|01>,
  |1>⊗|0>=|10>, |1>⊗|1>=|11>とすれば、
- |q1> ⊗ |q2> = (a|0> + b|1>) ⊗ (c|0> + d|1>)
  = ac|00>+ad|01>+bc|10>+bd|11>

- 二つのqubitの状態は、4つの状態の重ね合わせであることがわかる。



三個のqubitの状態

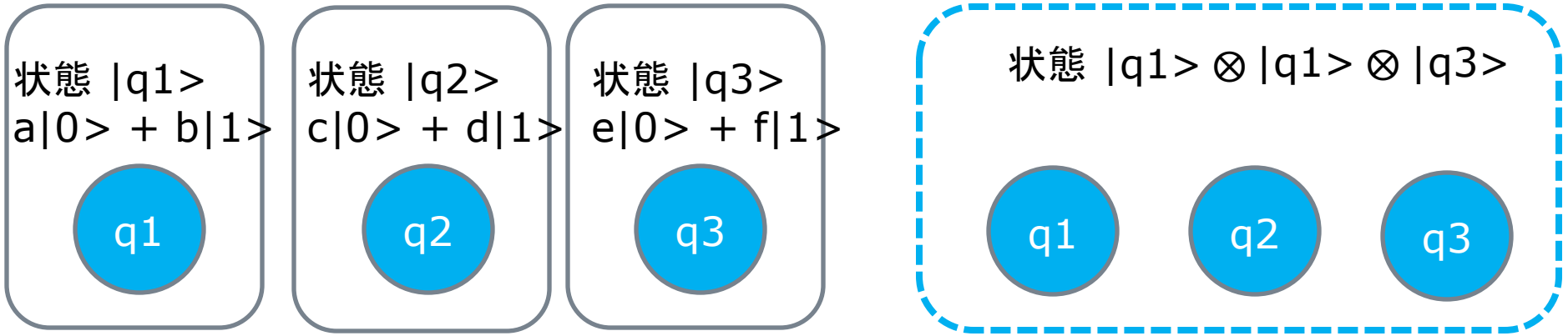
- 三つのqubit q1とq2とq3があったとする。
- q1,q2,q3の状態は、それぞれ、次のように表される。
  $|q1\rangle = a|0\rangle + b|1\rangle$
  $|q2\rangle = c|0\rangle + d|1\rangle$
  $|q3\rangle = e|0\rangle + f|1\rangle$
- この時、三つのqubit q1とq2とq3を、一緒に考えた状態を考えて、それを、状態 $|q1\rangle \otimes |q2\rangle \otimes |q3\rangle$と表すことにする。



三個のqubitの状態の計算

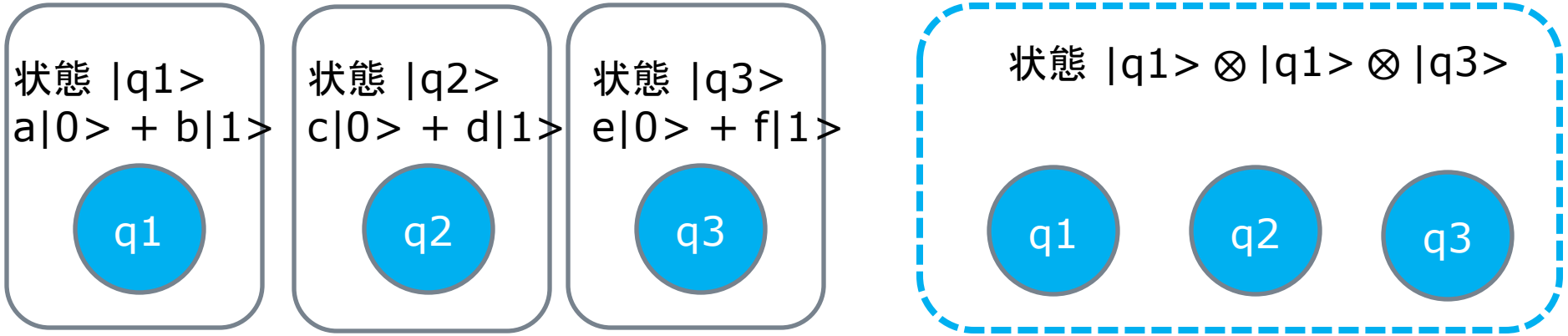
- この時、|q1> ⊗ |q2> ⊗ |q3>を、次のように計算する。

$$|q1\rangle \otimes |q2 \otimes |q3\rangle$$
$$= (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle) \otimes (e|0\rangle + f|1\rangle)$$



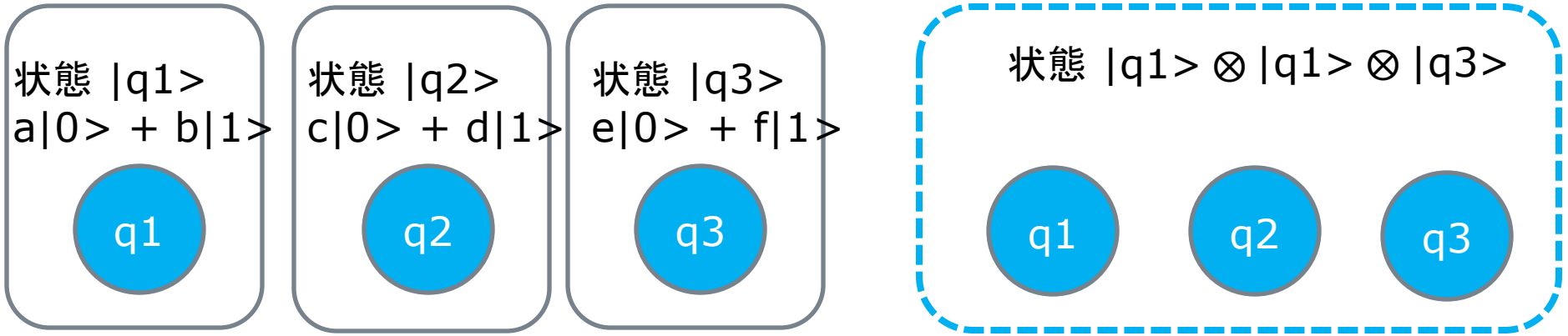
三個のqubitの状態の計算

- 先の式で、
  |0>⊗|0>⊗|0>=|000>, |0>⊗|1>⊗|0>=|010>,
  |1>⊗|0>⊗|0>=|100>, |1>⊗|1>⊗|1>=|111> …とすれば、
- |q1> ⊗ |q2 ⊗ |q3>
  = (a|0> + b|1>) ⊗ (c|0> + d|1>)⊗ (e|0> + f|1>)
  = ace|000>+acf|001>+ade|010>+adf|011>
  +bce|100>+bcf|101>+bde|110>+bdf|111>



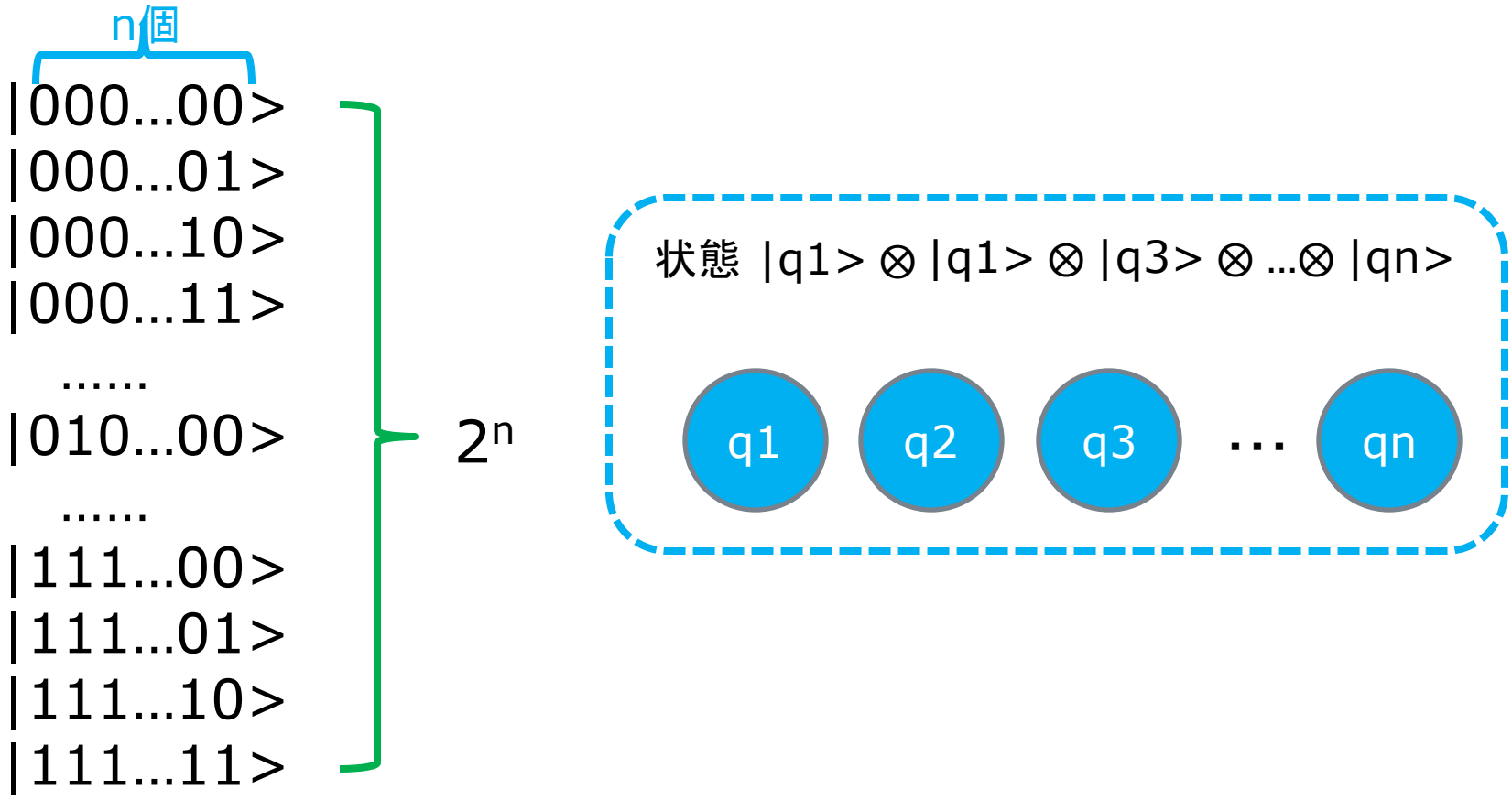
三個のqubitの状態の数

- よって、三個のqubitの状態は、
  $|q1\rangle \otimes |q2 \otimes |q3\rangle$
  $= (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle) \otimes (e|0\rangle + f|1\rangle)$
  $= ace|000\rangle + acf|001\rangle + ade|010\rangle + adf|011\rangle$
  $+ bce|100\rangle + bcf|101\rangle + bde|110\rangle + bdf|111\rangle$
  となって、
  $|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle$
  の8つの状態の重ね合わせになる。



n個のqubitの状態の数

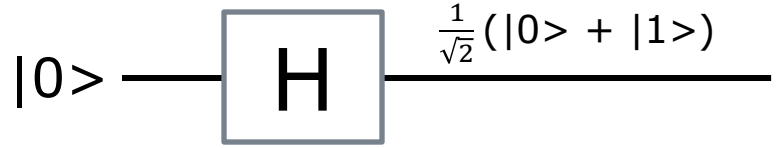
- 一般に、n個のqubitの状態は、
  2^n個の状態の重ね合わせになる



アダマールゲート H を並列に配置する

アダマールゲートをn個並列に配置した回路の出力を考える。

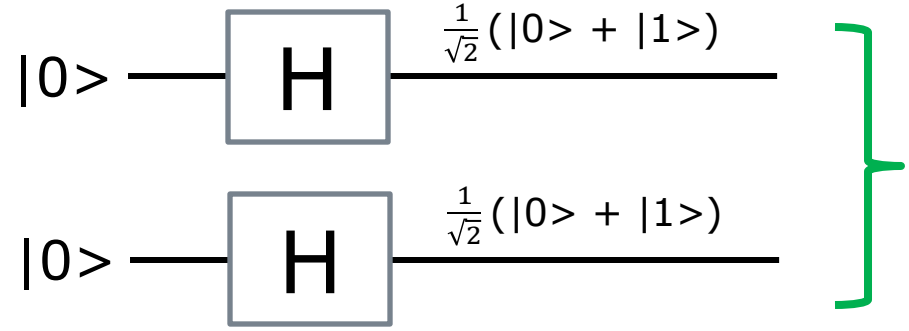
n=1



2個の基底

$\frac{1}{\sqrt{2}}(|0> + |1>)$

n=2



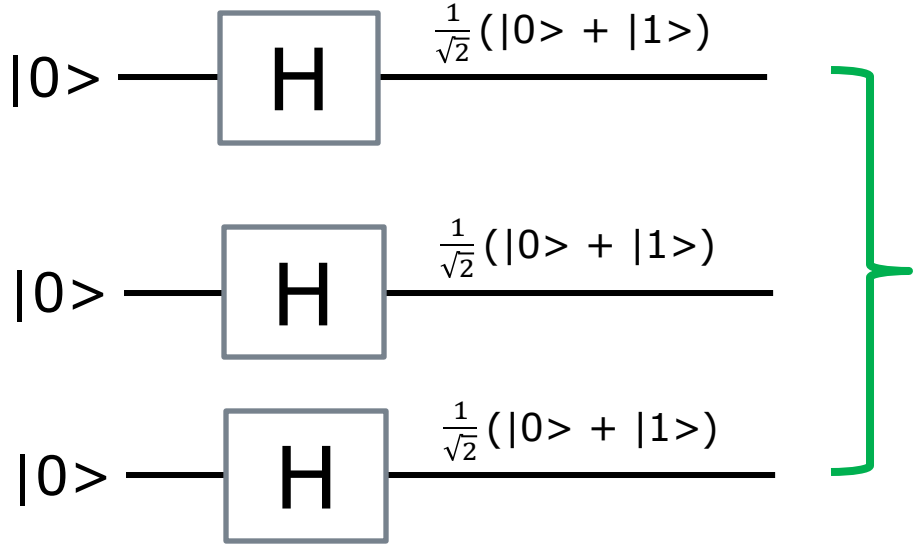
4個の基底

$\frac{1}{\sqrt{2^2}}$ (|00> + |01>
+|10> + |11>)

アダマールゲート H を並列に配置する

アダマールゲートをn個並列に配置した回路の出力を考える。

n=3

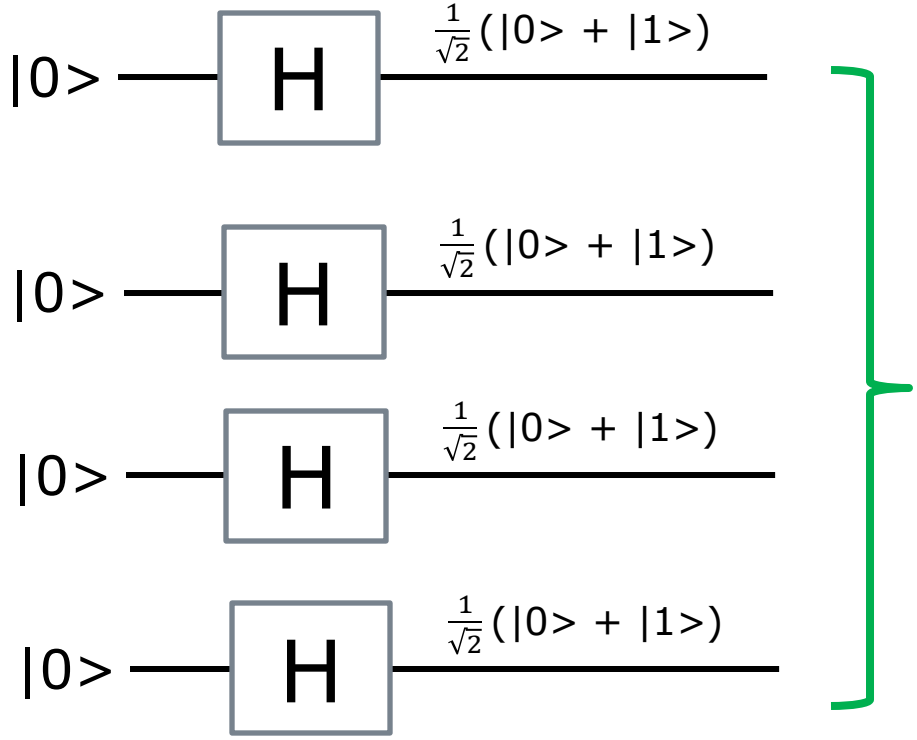


$$\frac{1}{\sqrt{2^3}}(|000> + |001> + |010> + |011> + |100> + |001> + |110> + |111>)$$

アダマールゲート H を並列に配置する

アダマールゲートをn個並列に配置した回路の出力を考える。

n=4



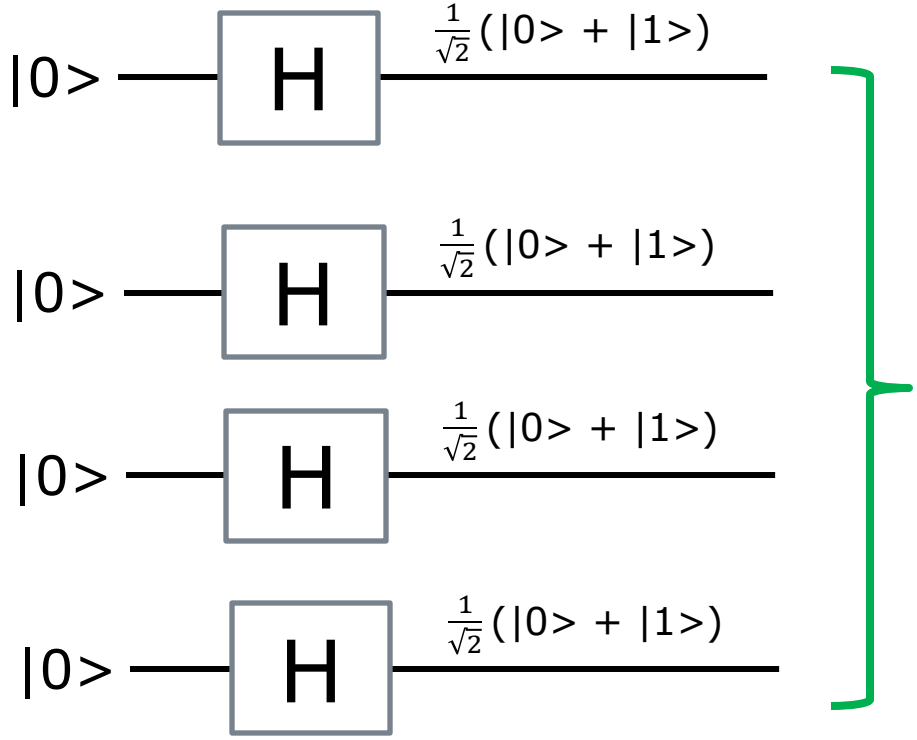
16個の基底

$$\frac{1}{\sqrt{2^4}}\Big(|0000> + |0001> + |0010> + |0011> + |0100> + |0001> + |0110> + |0111> + |1000> + |1001> + |1010> + |1011> + |1100> + |1001> + |1110> + |1111>\Big)$$

アダマールゲート H を並列に配置する

アダマールゲートをn個並列に配置した回路の出力を考える。

n=4



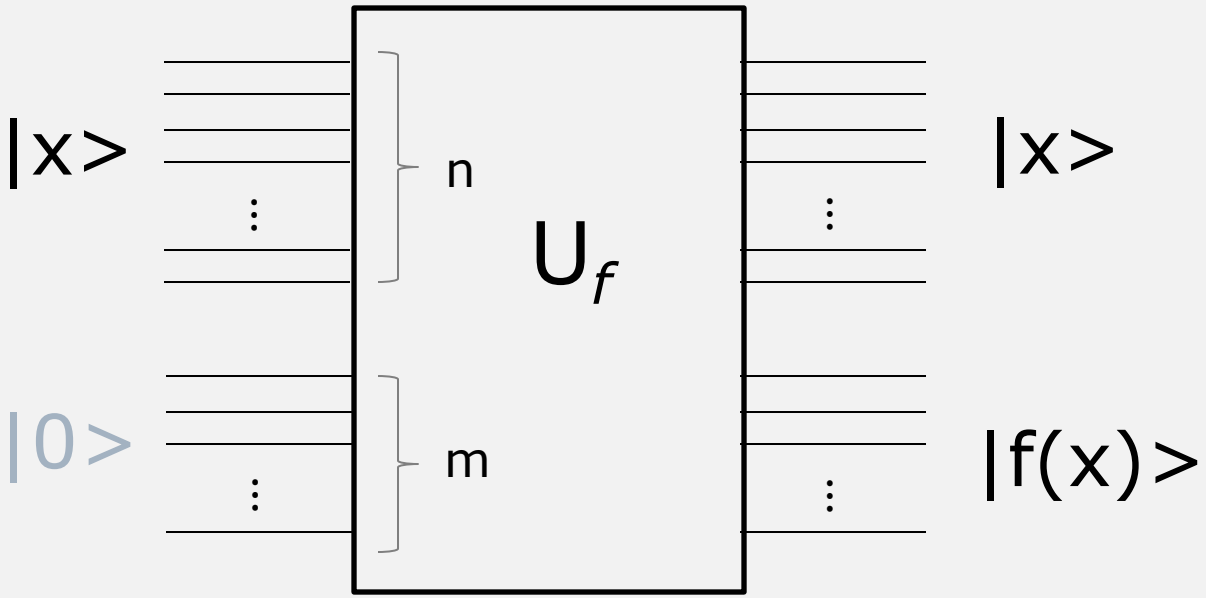
16個の基底

$$\frac{1}{\sqrt{2^4}}\Big(|0000> + |0001> + |0010> + |0011> + |0100> + |0001> + |0110> + |0111> + |1000> + |1001> + |1010> + |1011> + |1100> + |1001> + |1110> + |1111>\Big)$$

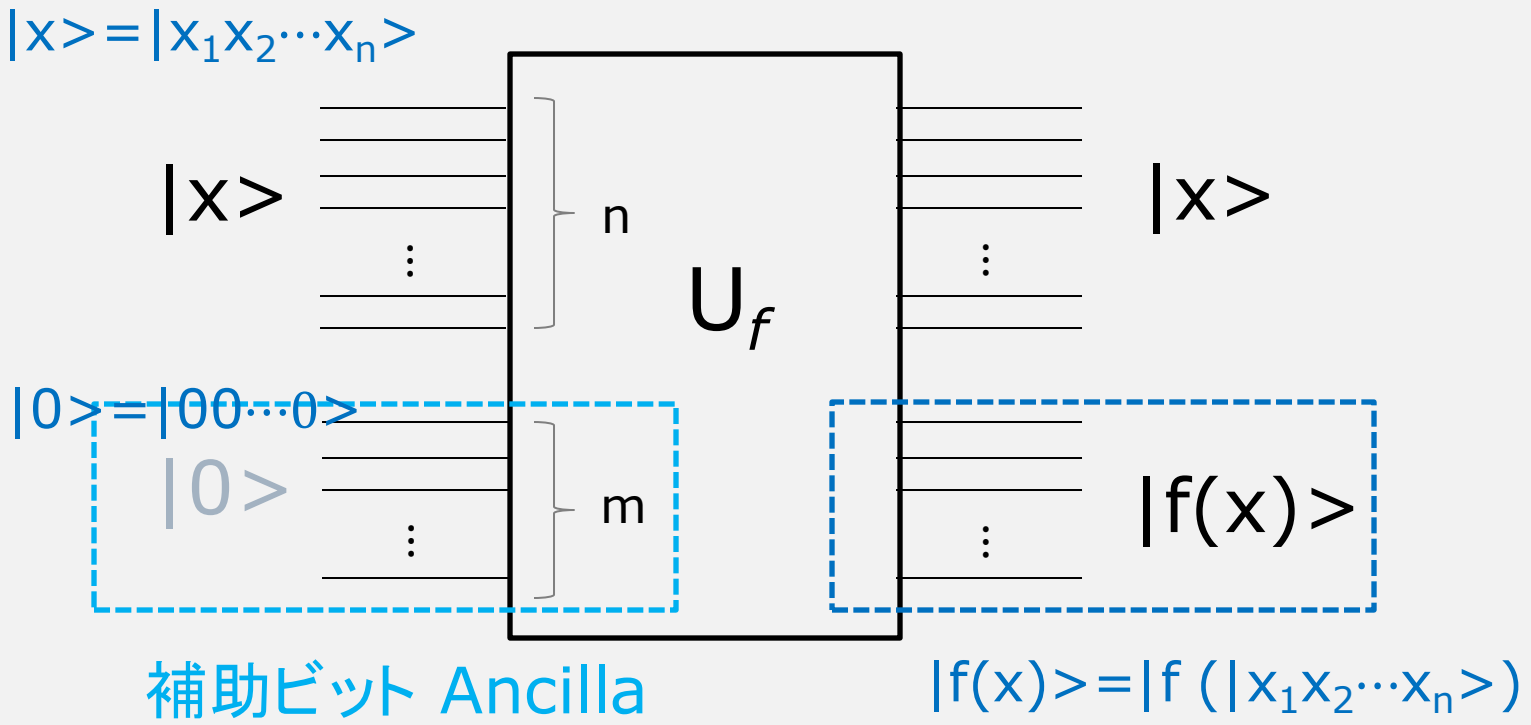
任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形

任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形

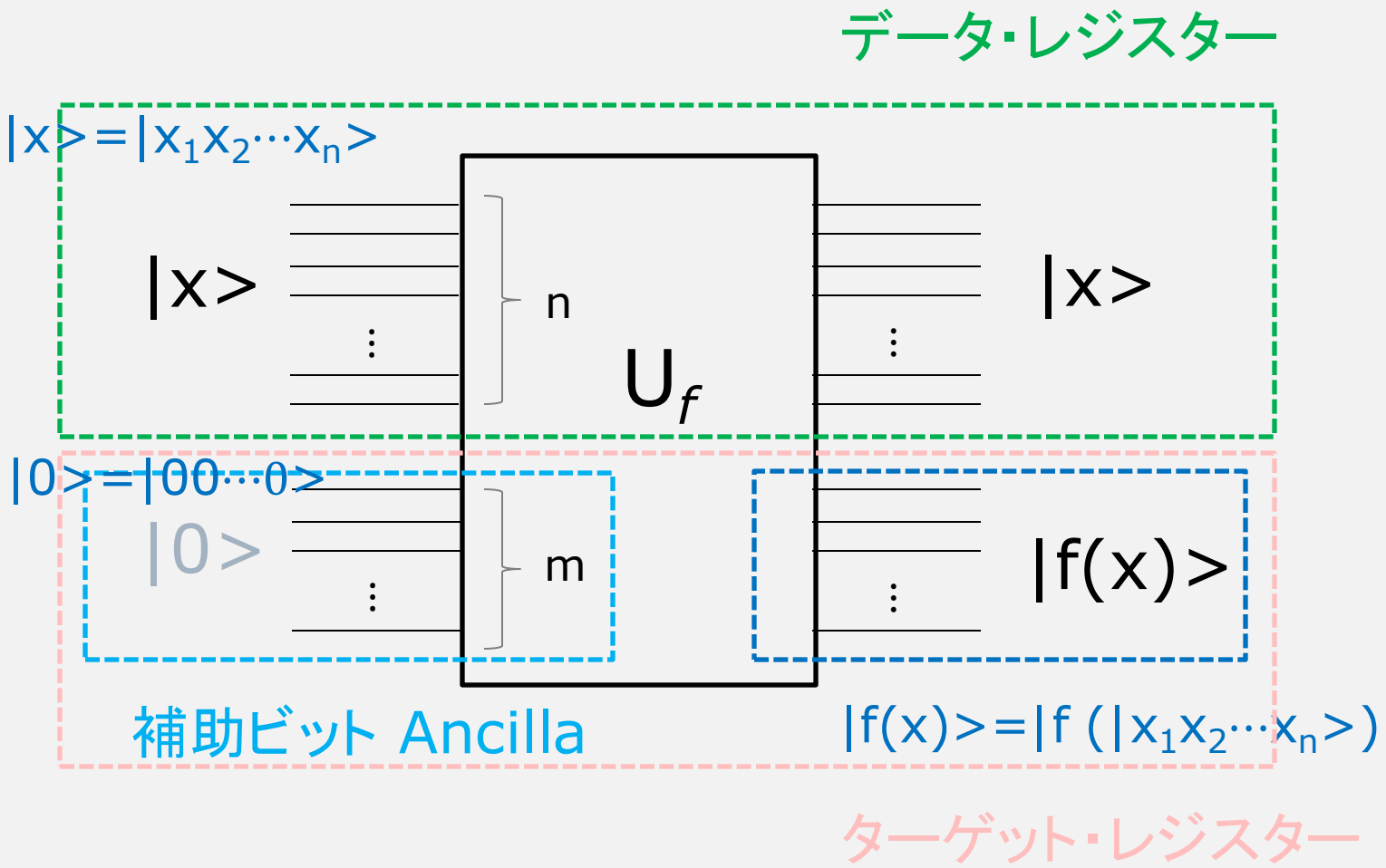
f(x)の具体的な実装を与えているわけではない。
ブラックボックスとかOracleと言ったりする



任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形

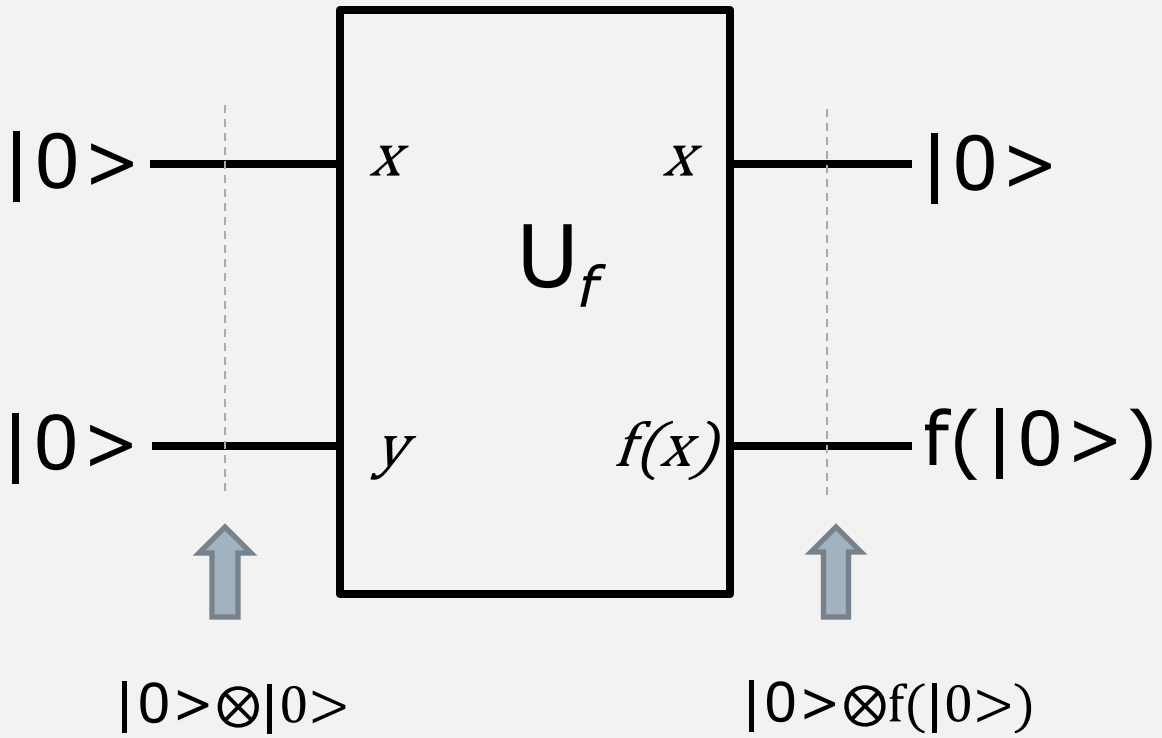


任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形



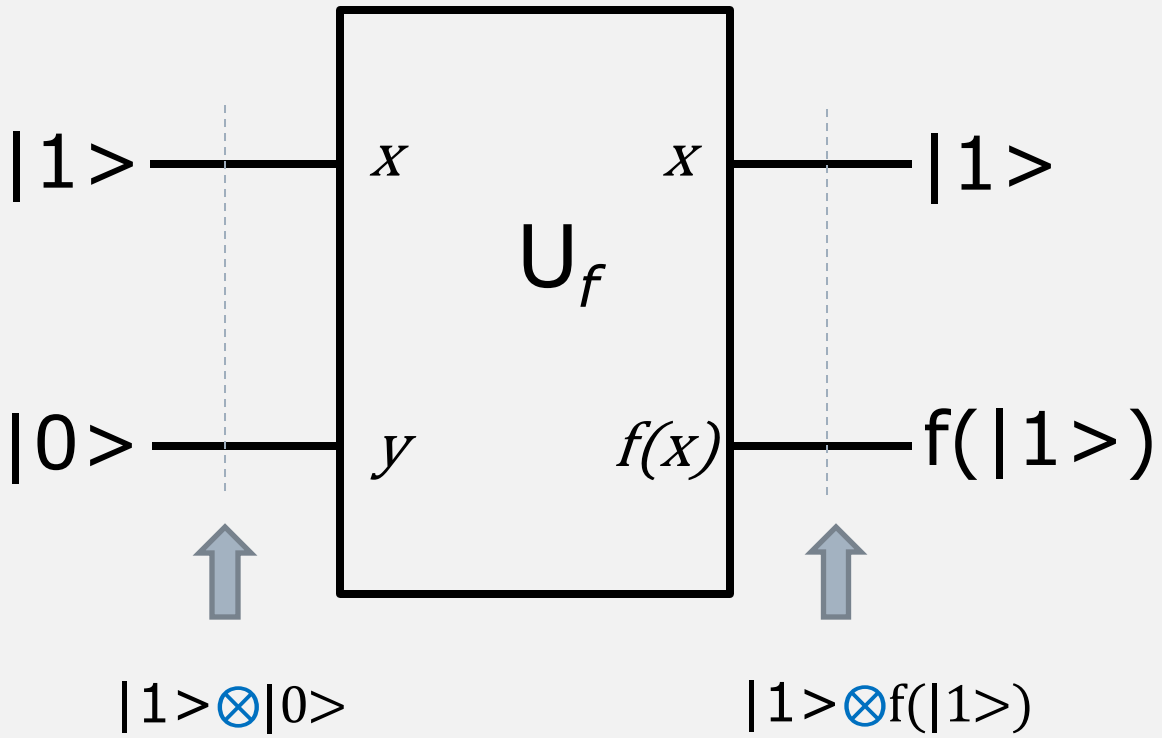
2-qubit(入力　1-qubit, 出力 1-qubit)の単純な回路U_fを考える

入力 x = |0> の時



2-qubit(fの入力 1-qubit, fの出力 1-qubit)の単純な回路U_fを考える

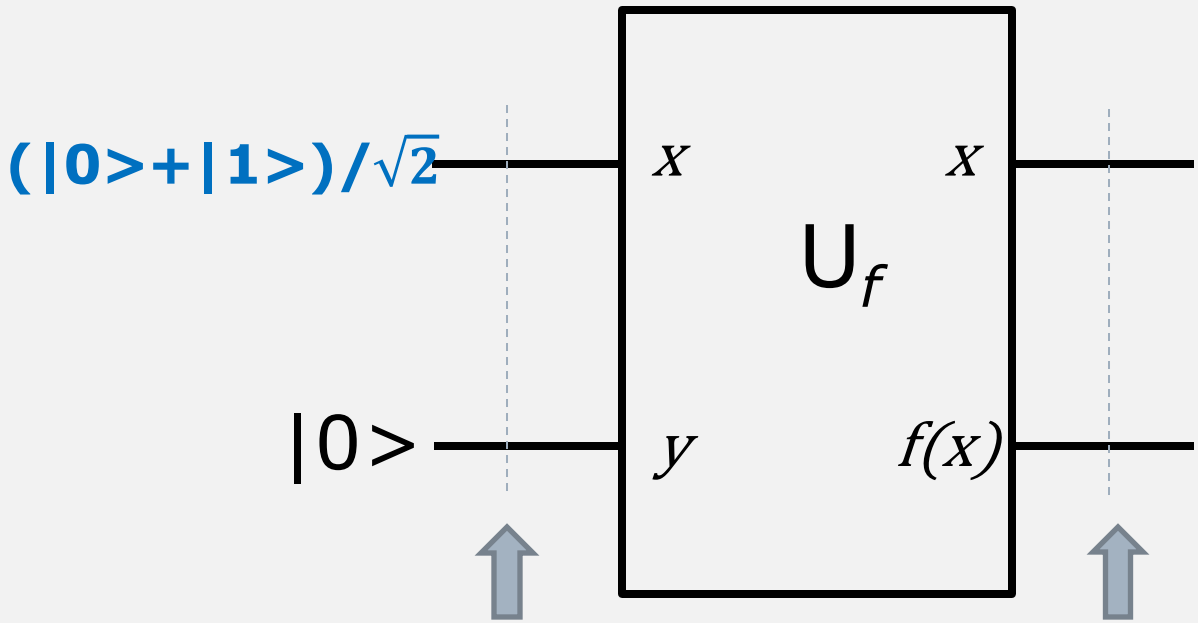
入力 x = |1> の時



意味が明確ならテンソル記号を省略してもいい

2-qubit(fの入力 1-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 $x = (|0>+|1>)/\sqrt{2}$ の時



回路の内部で
entangleが起きると
ライン毎に分離できない
可能性がある

ただ、Ufの出力全体
の状態は計算できる

$|(|0>+|1>)/\sqrt{2} \otimes |0>$

$|0>f(|0>)/\sqrt{2} + |1>f(|1>)/\sqrt{2}$

テンソル記号を省略した

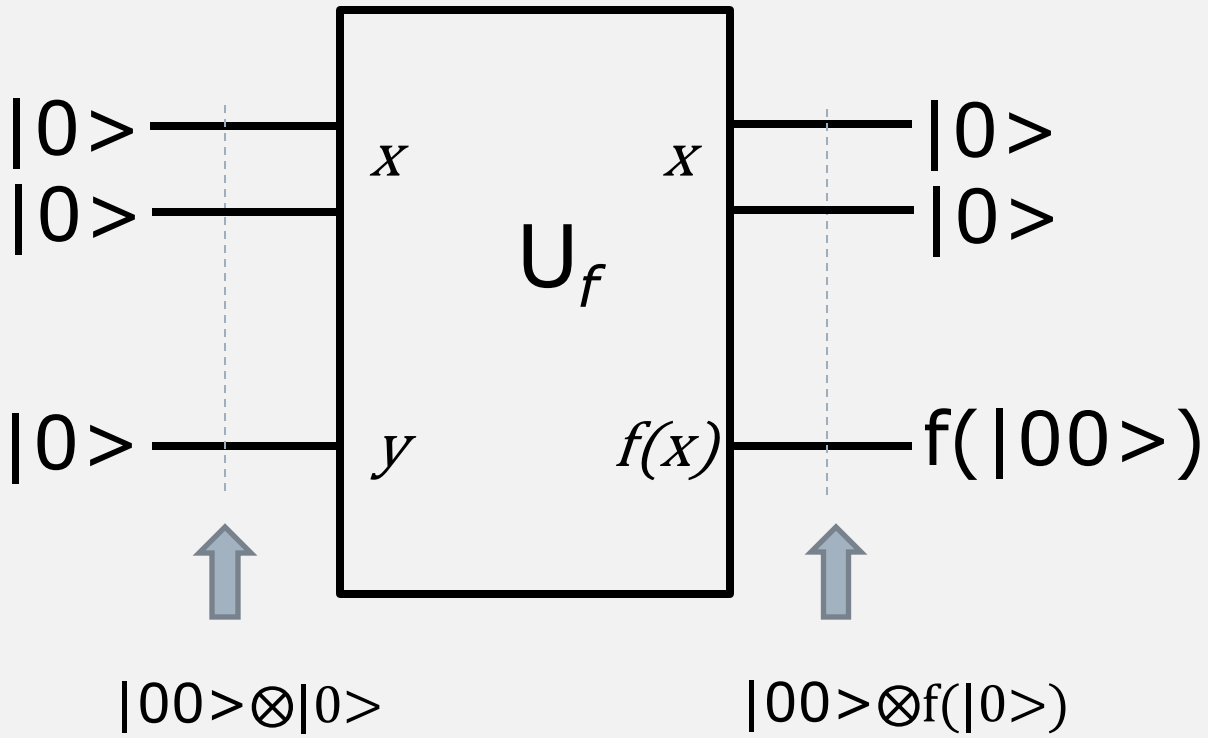
なぜ？

$|0\rangle f(|0\rangle)/\sqrt{2} + |1\rangle f(|1\rangle)/\sqrt{2}$

- 関数 f は線形であるとは限らないが、Ufはユニタリであり線形である。Mが線形であるとは、
  $M(a|A\rangle+b|B\rangle)=aM|A\rangle+bM|B\rangle$が成り立つことをいう。
- $x = |0\rangle$ の時と $x = |1\rangle$ の時の例から、Ufは次の式を満たすことがわかる。
  $Uf(|0\rangle\otimes|0\rangle)= |0\rangle\otimes f(|0\rangle)$
  $Uf(|1\rangle\otimes|0\rangle)= |1\rangle\otimes f(|1\rangle)$
- この時、Ufは線形であるので、
  $Uf((|0\rangle+|1\rangle)/\sqrt{2}\otimes|0\rangle)$
  $= Uf((|0\rangle/\sqrt{2}\otimes|0\rangle+|1\rangle/\sqrt{2}\otimes|0\rangle)$
  $= Uf(|0\rangle\otimes|0\rangle)/\sqrt{2} + Uf(|1\rangle\otimes|0\rangle)/\sqrt{2}$
  $= |0\rangle\otimes f(|0\rangle)/\sqrt{2} + |1\rangle\otimes f(|1\rangle)/\sqrt{2}$
  $= |0\rangle f(|0\rangle)/\sqrt{2} + |1\rangle f(|1\rangle)/\sqrt{2}$

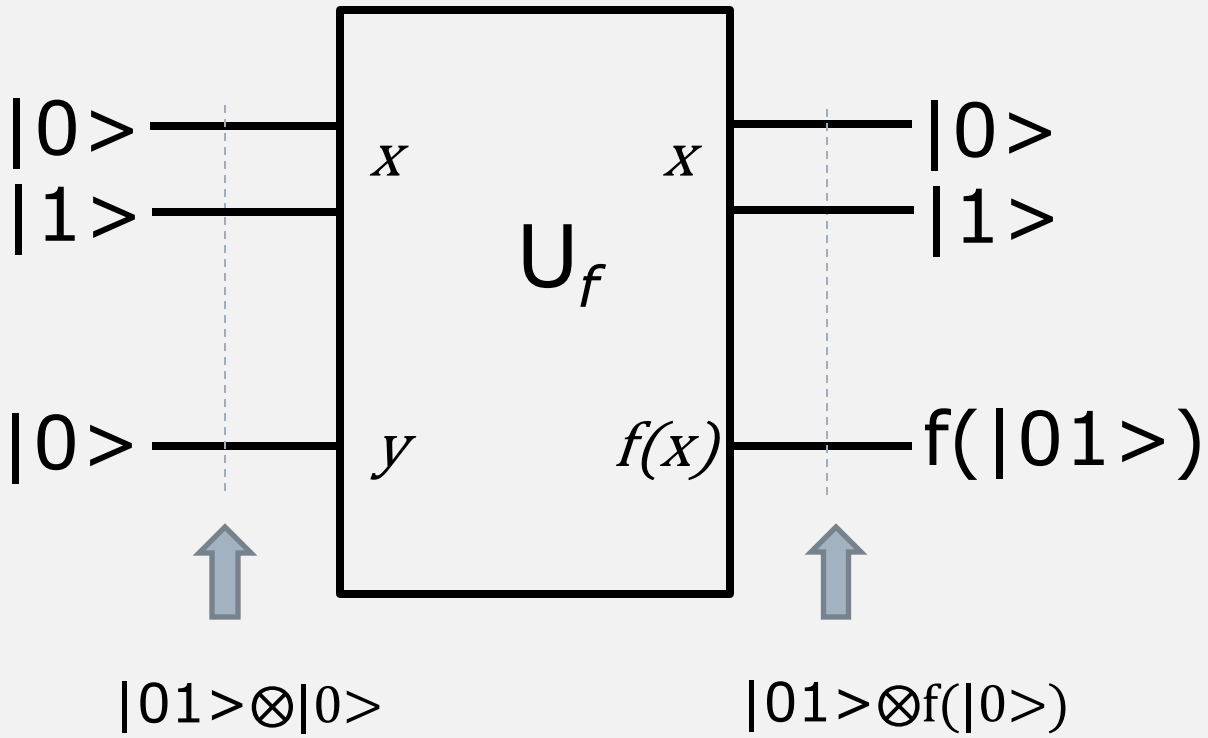
3-qubit(fの入力 2-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 x = |0>⊗|0> = |00> の時



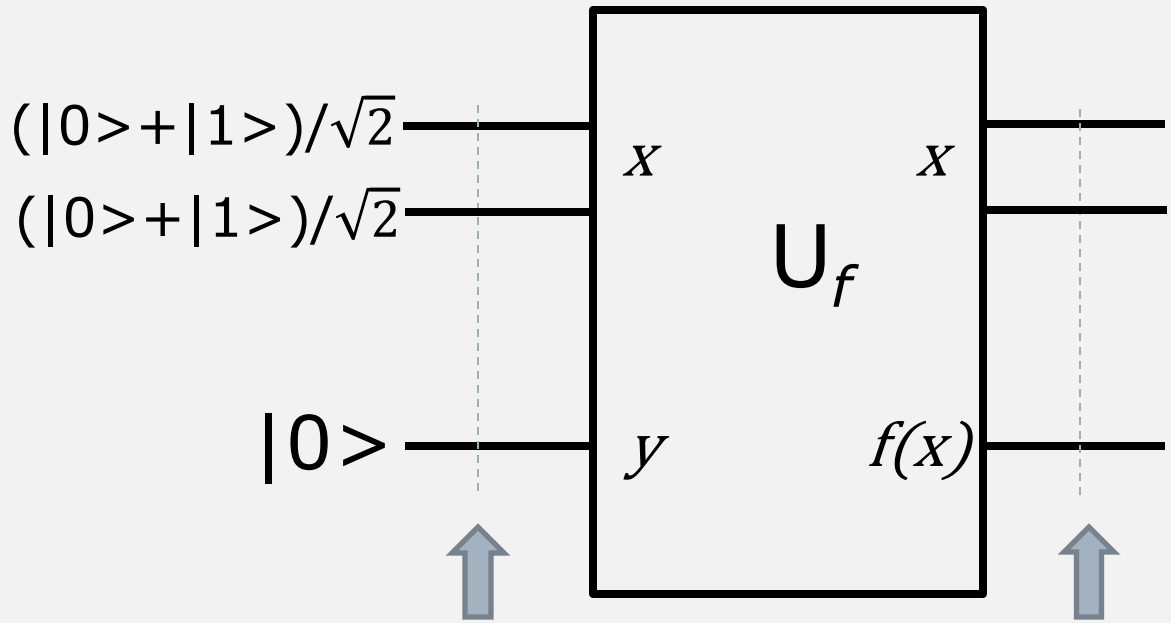
3-qubit(fの入力 2-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 x = |0>⊗|1> = |01> の時



3-qubit(fの入力 2-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 $x = (|0>+|1>)/\sqrt{2})\otimes(|0>+|1>)/\sqrt{2}$ の時



回路の内部で
entangleが起きると
ライン毎に分離できない
可能性がある

ただ、Ufの出力全体
の状態は計算できる

|x>⊗|0>

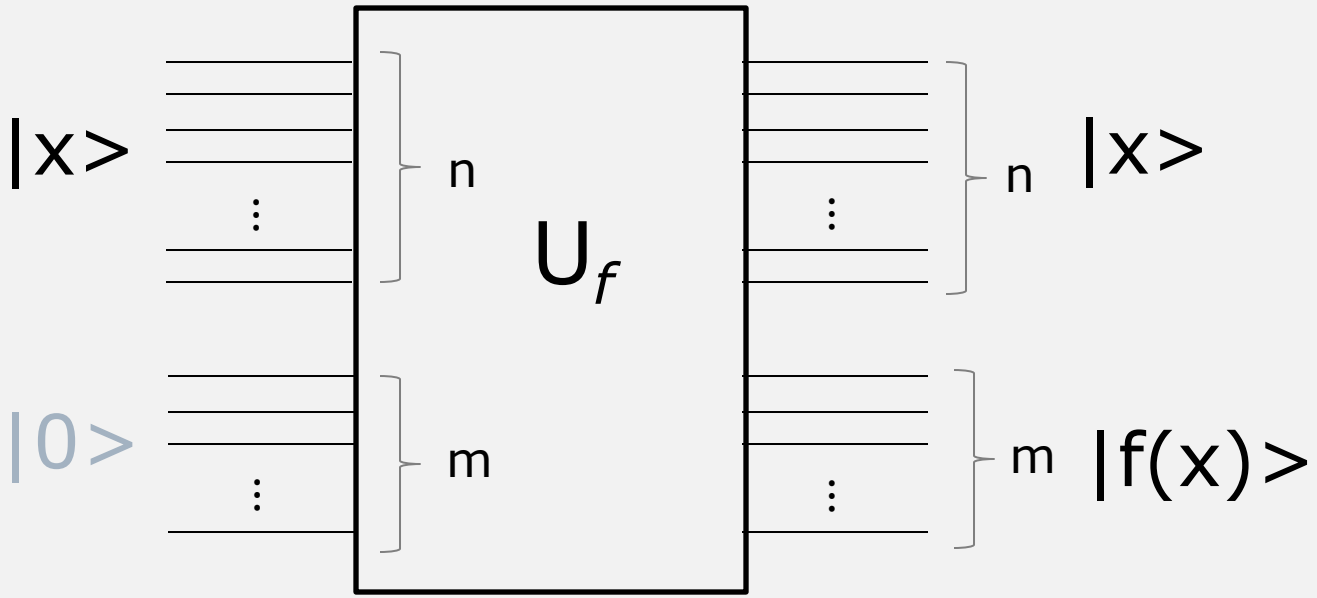
(|00>f(|00>) + |01>f(|01>) +
|10>f(|10>) + |11>f(|11>))/2

なぜなら

- $x = (|0\rangle+|1\rangle)/\sqrt{2})\otimes(|0\rangle+|1\rangle)/\sqrt{2}$ とする。
  $x = (|00\rangle+|01\rangle+|10\rangle+|11\rangle)/2$ である。
- この時、

$$
\begin{aligned}
&Uf(|x\rangle\otimes|0\rangle)\\
&= Uf((|00\rangle+|01\rangle+|10\rangle+|11\rangle)\otimes|0\rangle)/2\\
&= (Uf(|00\rangle\otimes|0\rangle)+Uf(|01\rangle\otimes|0\rangle)+\\
&\quad Uf(|10\rangle\otimes|0\rangle)+Uf(|11\rangle\otimes|0\rangle))/2\\
&= (|00\rangle\otimes f(|00\rangle)+|01\rangle\otimes f(|01\rangle)+\\
&\quad |10\rangle\otimes f(|10\rangle)+|00\rangle\otimes f(|11\rangle)/2\\
&=(|00\rangle f(|00\rangle) + |01\rangle f(|01\rangle) +\\
&\quad |10\rangle f(|10\rangle) + |11\rangle f(|11\rangle))/2
\end{aligned}
$$

ここまでは、f(x)が1-qubitで表現される例をみてきた。このことは、関数 f(x)が、0または1の値をとることを意味する。
もし、関数f(x)が m-qubitで表現されるのなら、そのことは関数 fが、0～2^{m-1}の範囲の値を取ることを意味する。それは、古典ビットでも同様である。こうした一般化のもとでも、先に見たQuantum Parallelismは可能である。



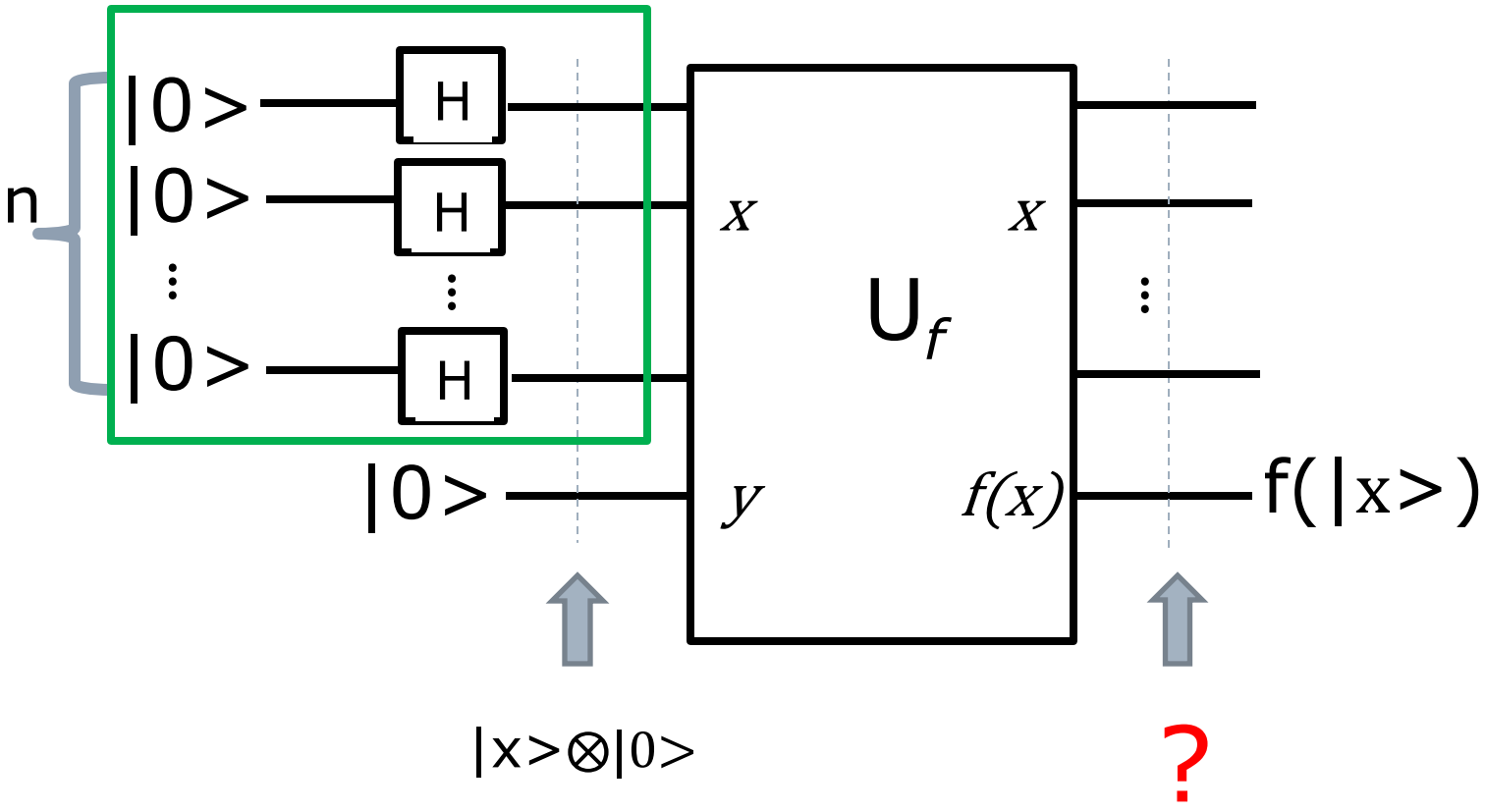
$\{0,1\}^n \ni x$ で、

Uf($\Sigma|x>|0>$) = Σ Uf ($|x>|0>$) = Σ $|x>|f(x)>$

Quantum Parallelism

(n+1)-qubit(fの入力　n-qubit, fの出力 1-qubit)の回路U_fを考える

Ufの入力に、パラレルに置かれたアダマール・ゲート n個の出力（それぞれの入力は|0>）をつなげてみよう



アダマール･ゲート n 個の出力

n個の出力のテンソル積

n個

$$\Rightarrow \frac{1}{\sqrt{2}}(|0>+|1>)\otimes\frac{1}{\sqrt{2}}(|0>+|1>)\otimes\cdots\otimes\frac{1}{\sqrt{2}}(|0>+|1>)$$

$$=\left(\frac{1}{\sqrt{2}}\right)^{n}\sum|Xi>$$

$|Xi>$は、次のようなすべての基底である

n桁

$$|X_0> = |000\cdots\cdots000>$$
$$|X_1> = |000\cdots\cdots001>$$
$$|X_2> = |000\cdots\cdots010>$$
$$|X_3> = |000\cdots\cdots011>$$
$$\vdots \qquad\qquad \vdots$$
$$|X_k> = |111\cdots\cdots111>$$

2^n個

先の回路U_fの出力を考える

- Ufへの入力xは、$H^{\otimes n}|0\rangle^{\otimes n}$ で、これは$\left(\frac{1}{\sqrt{2}}\right)^n \sum |Xi\rangle$ に等しい。ただし、$Xi \in \{0,1\}^n$であるすべてについて和をとる。
- $Uf(|x\rangle \otimes |0\rangle) = Uf\left(\left(\frac{1}{\sqrt{2}}\right)^n \sum |Xi\rangle \otimes |0\rangle\right)$

  $= \left(\frac{1}{\sqrt{2}}\right)^n \sum Uf(|Xi\rangle \otimes |0\rangle)$

  $= \left(\frac{1}{\sqrt{2}}\right)^n \sum (|Xi\rangle \otimes f(|Xi\rangle))$

Quantum Parallelism

- 例えば、n=3の時、Ufの出力は次のようになる。

$$\left(\frac{1}{\sqrt{2}}\right)^3 (\; |000>\mathrm{f}(|000>) + |001>\mathrm{f}(|001>) + |010>\mathrm{f}(|010>) + |011>\mathrm{f}(|011>) + |100>\mathrm{f}(|100>) + |101>\mathrm{f}(|101>) + |110>\mathrm{f}(|110>) + |111>\mathrm{f}(|111>) \;)$$

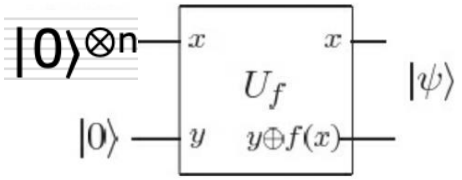
ただし、|x> ⊗ f(|x>)のテンソル記号を省略している。
三つのqubitに対する一回のUfの呼び出しが、f(|000>)から
f(|111>)までの8つのfの値を計算していることがわかる。

一般に、n個のqubitに対するUfの呼び出しは、2^n個の f の値を計算することになる。

Quantum Parallelism

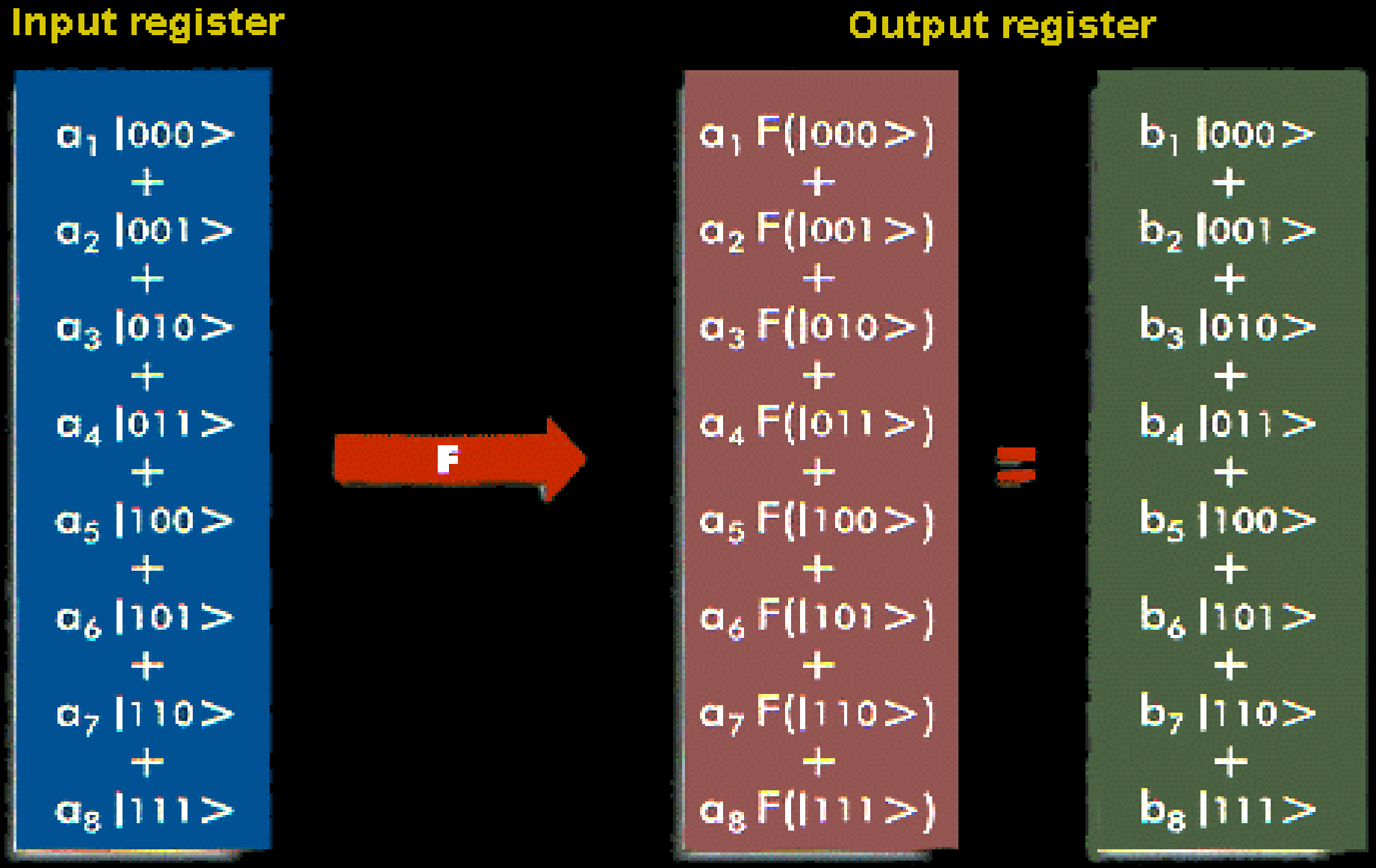
- nビットの入力xと1ビットの出力 f(x)を持つ関数の量子並列評価は、次のように実行される。
- n+1 qubitの状態 $|0\rangle^{\otimes n}|0\rangle$を用意する。次に、最初のn qubitに、アダマール変換を適用する。その出力をU_fの量子回路の入力に接続すると、次の状態が生み出される。

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle|f(x)\rangle$$



- ある意味では、量子並列処理は、関数fの全ての可能な2^n個の値を同時に評価する。たとえ、我々は明らかにfを一回だけ評価するのであるが。

n個のqubitで、2^n個の並行計算が可能?



$\Sigma|x> f(|x>)$の観測

- 重ね合わせの状態の$\Sigma|x> f(|x>)$ を観測したとたんに、重ね合わせの状態は失われ、我々が観測するのは 個別の $|x> f(|x>)$のみである。
- 例えば、n＝3の時、Ufの出力が次のようになっていても、

$$\left(\frac{1}{\sqrt{2}}\right)^3 (\\ |000> f(|000>) + |001> f(|001>) \\ + |010> f(|010>) + |011> f(|011>) \\ + |100> f(|100>) + |101> f(|101>) \\ + |110> f(|110>) + |111> f(|111>) \\ )$$

我々が観測できるのは、xの一つの値についての $|000> f(|000>)$ または $|001> f(|001>)$ または･･･ $|111> f(|111>)$ のみである。

何かが必要である

- 量子並列計算は、直ちに有用なわけではない。
  最初の単一qubitのサンプルで $\frac{|0, f(0)\rangle + |1, f(1)\rangle}{\sqrt{2}}$:、

  この状態の測定は、|0> f(0) 〉か |1> f(1)〉の値のどれかを返すだけだ。もっと一般的に、状態 Σ_x |x> f(x) 〉の測定は、一つの値xについてのf(x)の値を返すだけだ。もちろん、古典的なコンピュータは、そうしたことを簡単にやってのける。

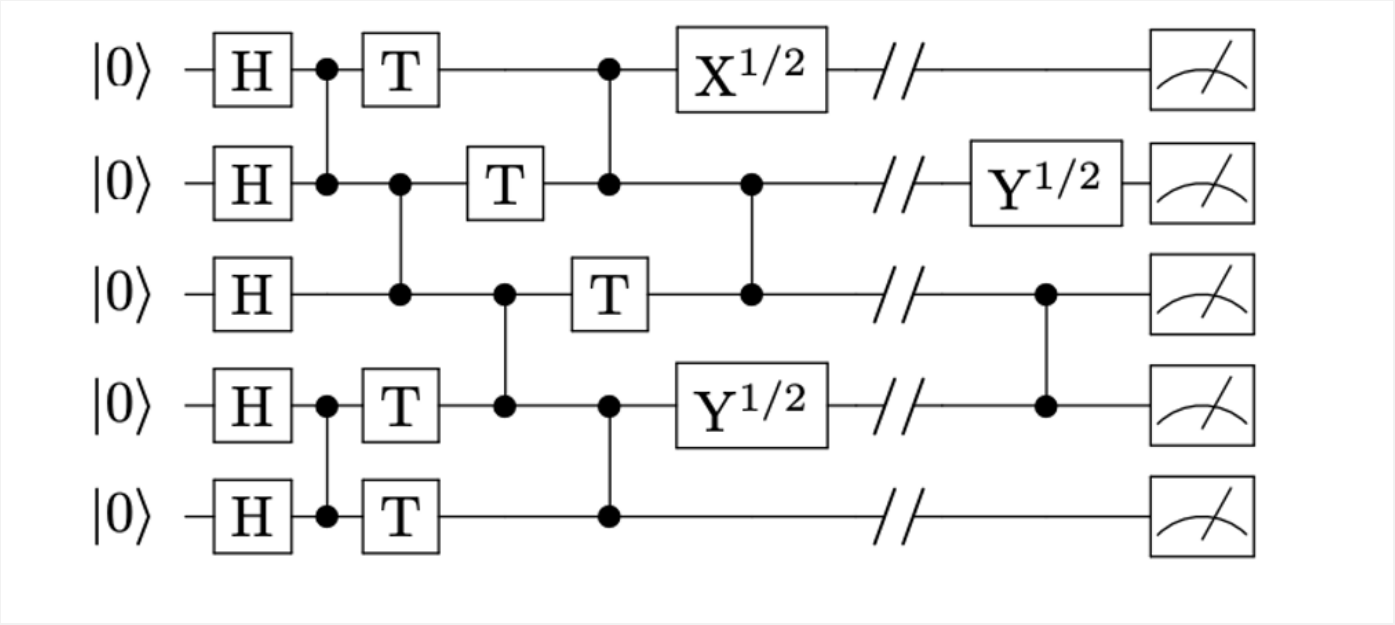
- 量子計算が役に立つためには、単なる量子並行計算以上の何かが必要になる。すなわち、Σ_x |x> f(x) 〉のような重ね合わせの状態から、f(x)の一つの値より多くの値の情報を引き出せるような能力が必要になる。

ランダム量子回路

Googleの実験の一つのポイントは、量子優越性を実証するのに、「ランダム量子回路」という手法をとったことである。ここでは、それがどういうアイデアかを説明する。

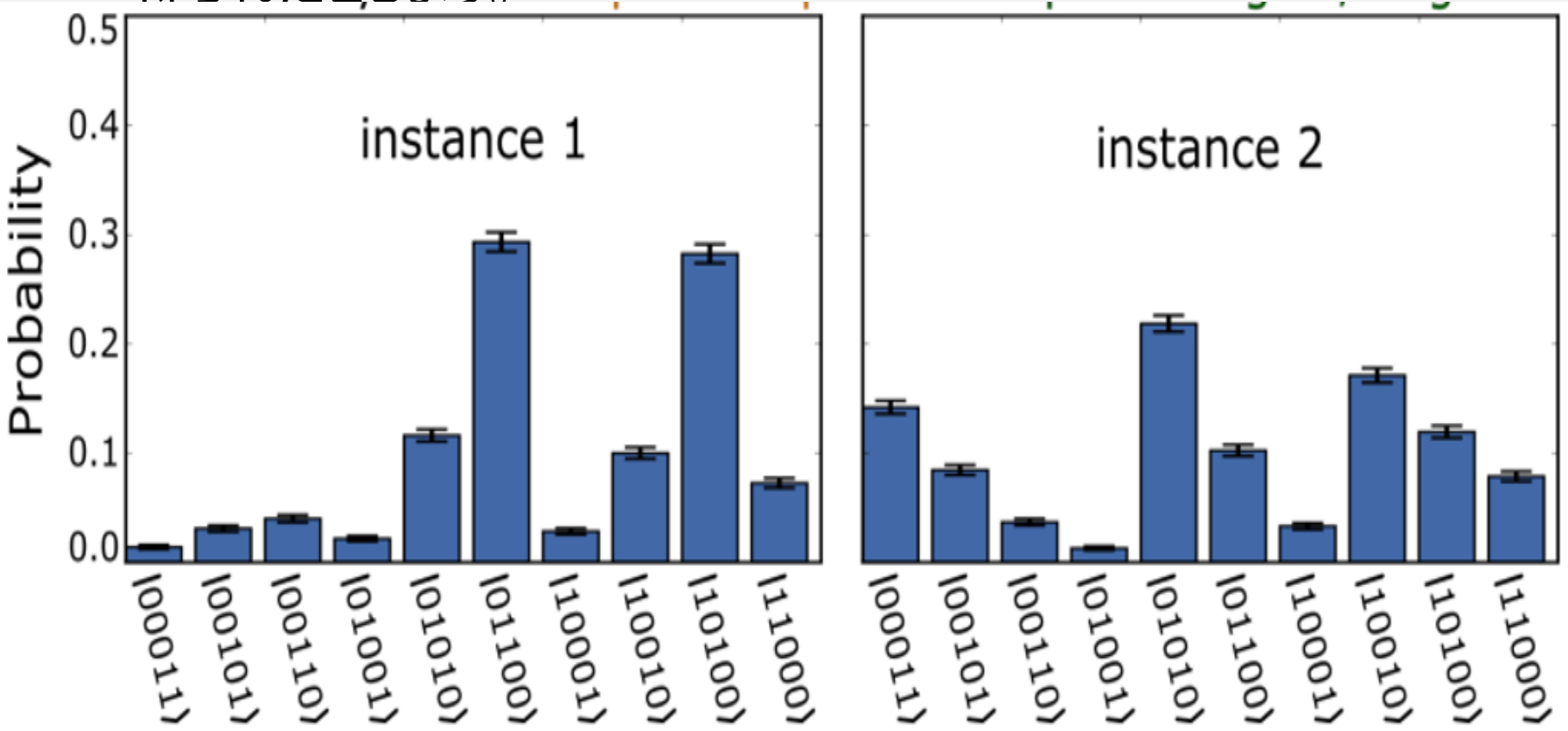
ランダムに量子回路を生成する

- ランダムに量子回路を生成する。この回路が何を計算しているかは考えない。
- 二つの別のランダム量子回路を、インスタンス1とインスタンス2としよう。



ランダム量子回路の出力をサンプリングし、出力の分布をチェックする

- 二つの量子回路の出力をサンプリングして、次のような分布が得られたとしよう。



得られた分布は、回路の特徴を反映している

- インスタンス1の回路と、インスタンス2の回路は、どちらもランダムに作られたものだが、それぞれ異なった回路である。その回路の違いが、分布の違いに反映している。
- サンプリングの数を増やしていけば、それぞれの回路に固有な分布の特徴は、いっそうはっきりしたものになってゆくだろう。

回路図が与えられれば、コンピュータを使って出力をシミレートできる

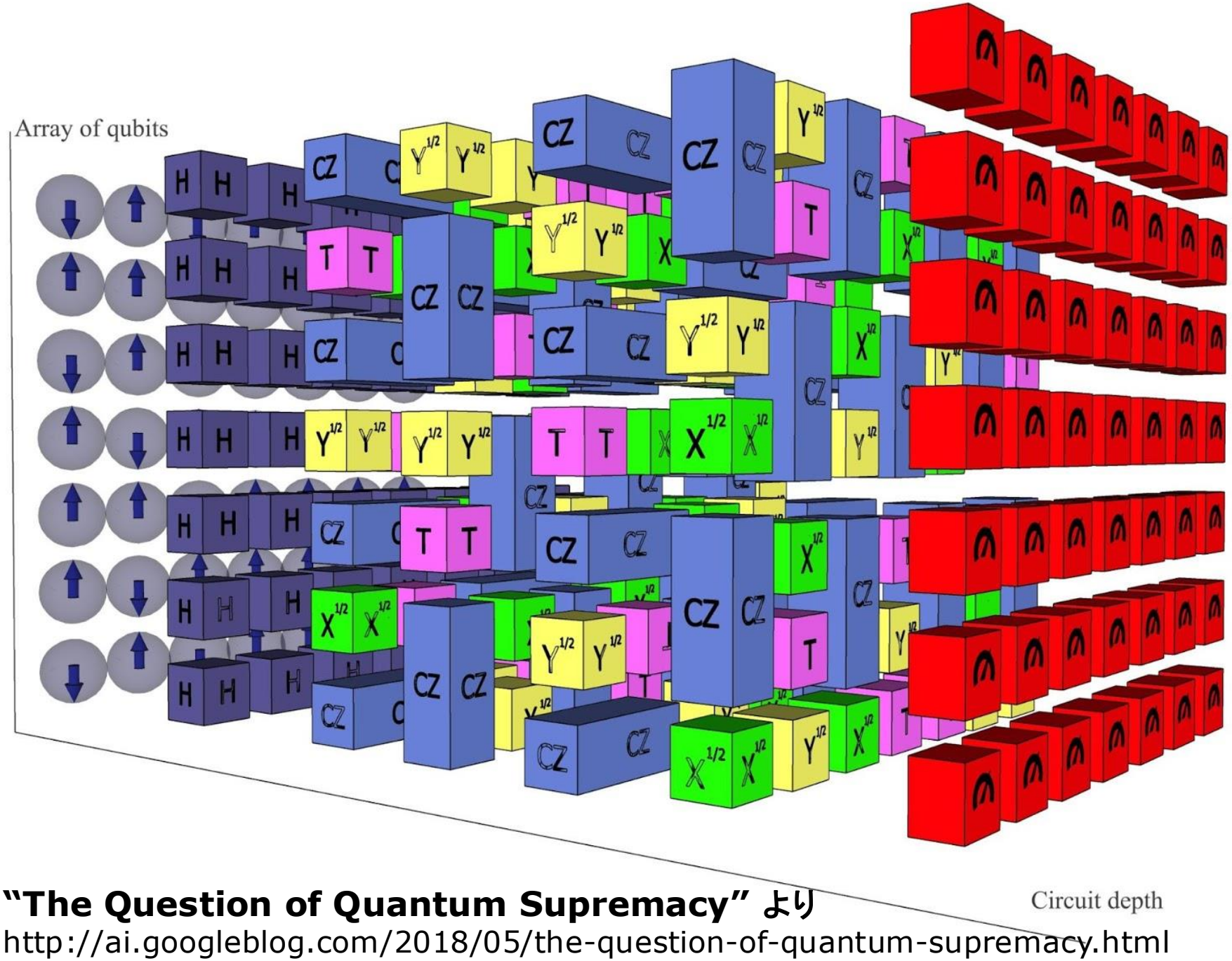
- もしも、インスタンス1とインスタンス2の回路図が与えられれば、コンピュータを使って、その出力をシミレートでき、サンプリングの数を増やせば、量子回路を使わなくても、正確な分布を得ることができるだろう。

- 問題は、これからである。
  量子回路の出力のサンプリングで作られた分布と、コンピュータの回路シミレートのサンプリングで作られた分布は、サンプル数を増やせば、基本的に同じものになるはずである。
- 量子回路もコンピュータでのシミレーションも、基本的には、「同じ仕事」をしたと考えることができる。それでは、この同じ仕事に要した時間は、それぞれ、どれくらいかかるのだろう？

ランダム量子回路を使った、量子優越性の実証

- 今回のGoogleの実験は、ランダム量子回路の出力を直接サンプリングする方が、コンピュータを使ってシミレートするよりも、圧倒的に速いことを示そうとしたものである。

- 実際、実験では、53qubit x 20段（これを「深さ」という）上の量子回路の100万回のサンプリングを 200秒で終えた。
- スーパーコンピュータが、この回路のシミレーションを行おうとすると、膨大な時間がかかる。論文では、それを「1万年」と見積もったが、そこは違っていたようだ。IBMの見積もりによると、「2.5日」だという。



“The Question of Quantum Supremacy” より

http://ai.googleblog.com/2018/05/the-question-of-quantum-supremacy.html

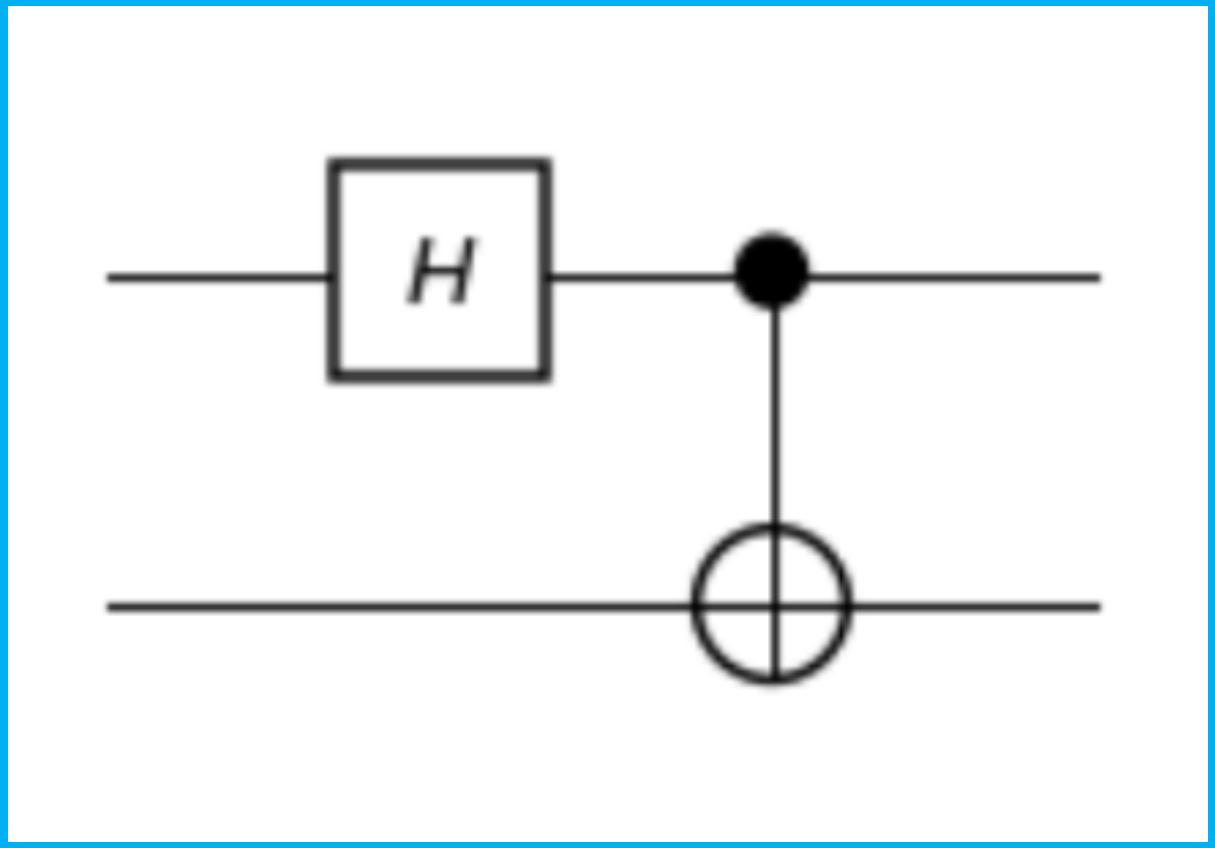
エンタングルメントと量子通信

参考資料
「量子通信入門」
https://www.marulabo.net/docs/teleportation-2/

エンタングルメントと量子通信

- エンタングルメント状態をつくる回路
- Bell State GateとBell Measure Gate
- 量子通信
  - Superdense Coding
  - 量子テレポーテーション
  - Entanglement Swapping

エンタングルメント状態をつくる回路

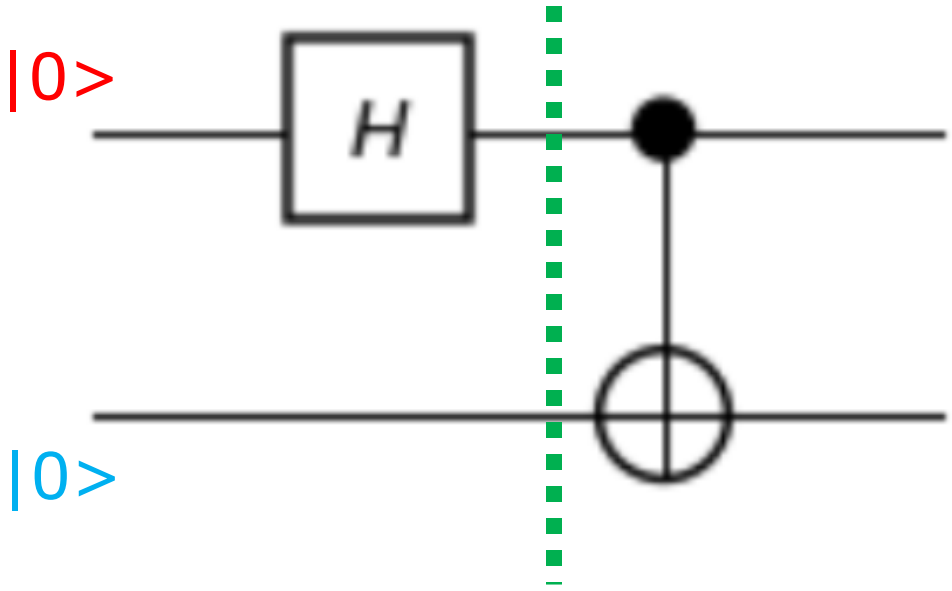


Bell State ゲート

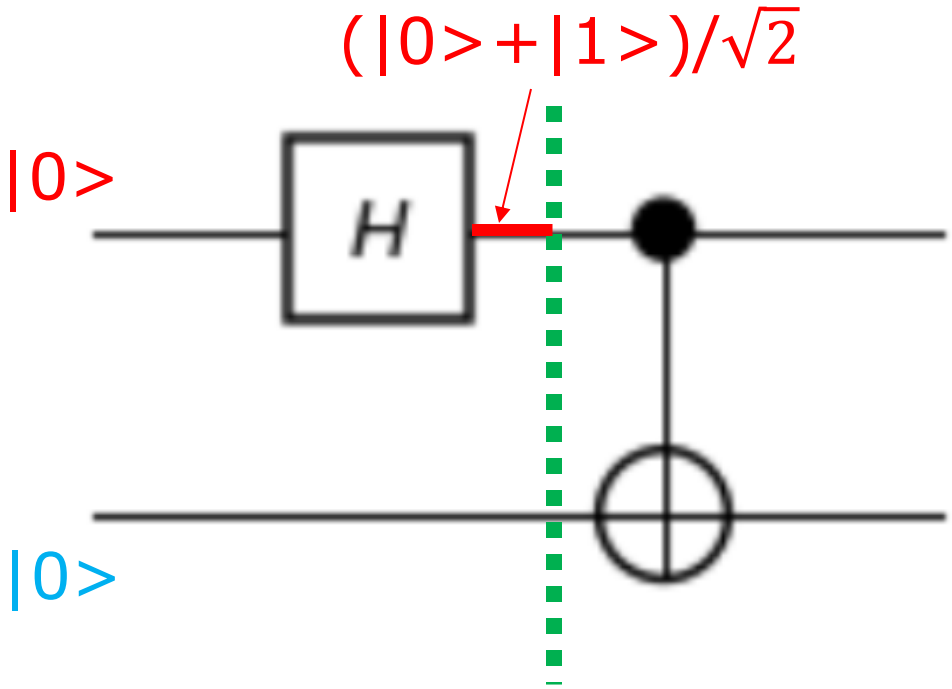
Bell State ゲートの働き

入力|00>の場合

この時点での
系全体の状態
を調べる

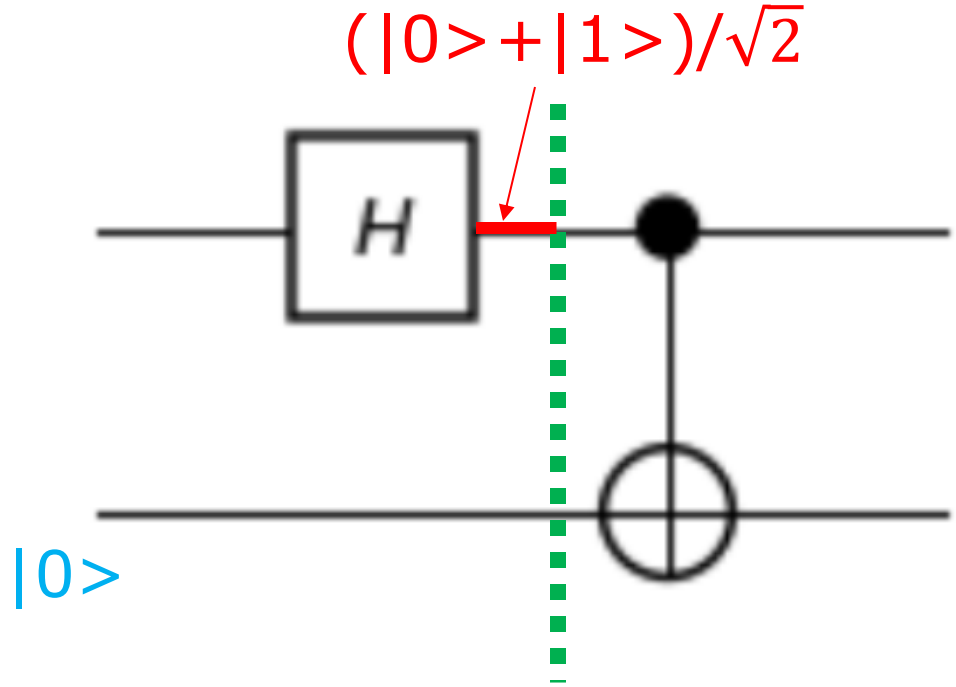


Bell State ゲートの働き
入力|00>の場合



Hは|0>を
(|0>+|1>)/√2に
変える

Bell State ゲートの働き
入力|00>の場合

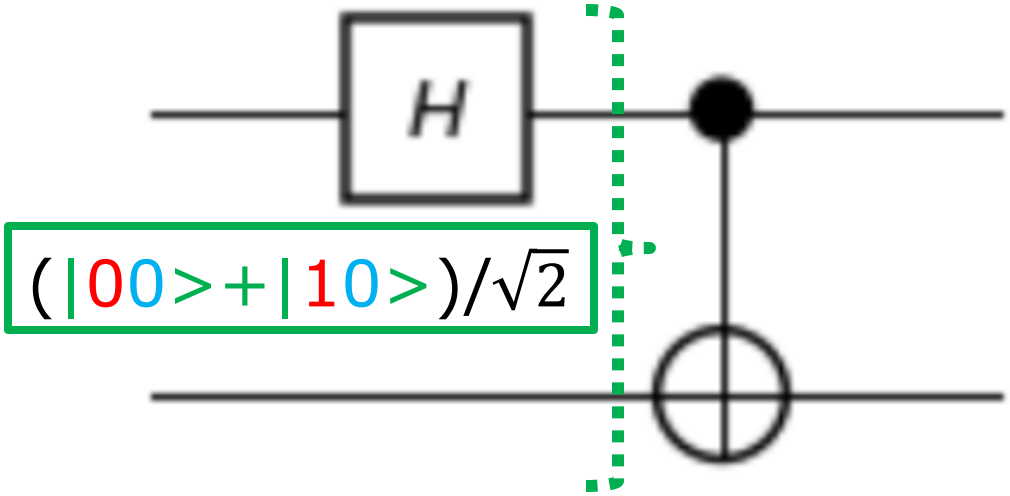


この時点での
系全体の状態

$$= (|00\rangle + |10\rangle)/\sqrt{2}$$

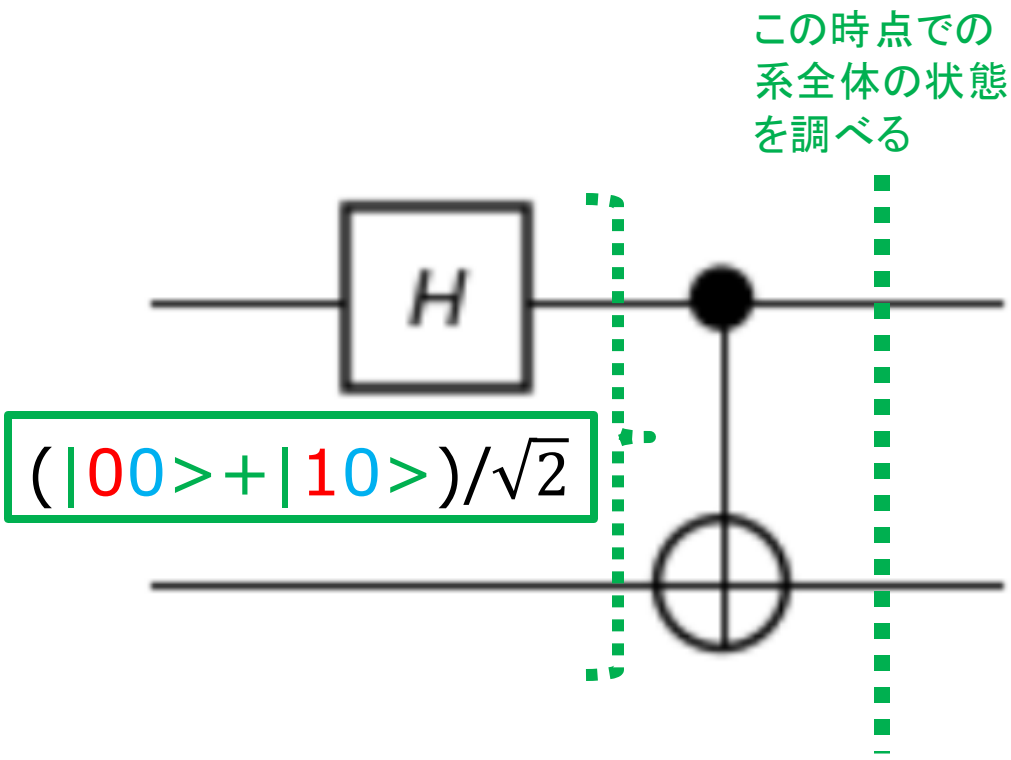
Bell State ゲートの働き

入力|00>の場合



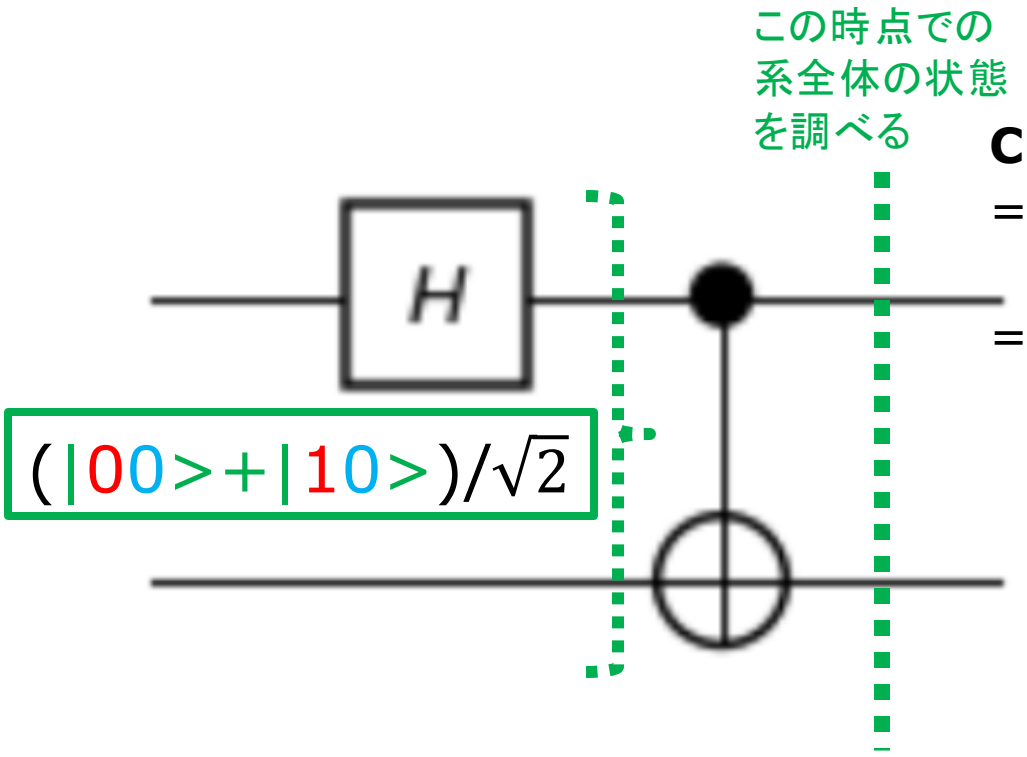
Bell State ゲートの働き

入力|00>の場合



Bell State ゲートの働き

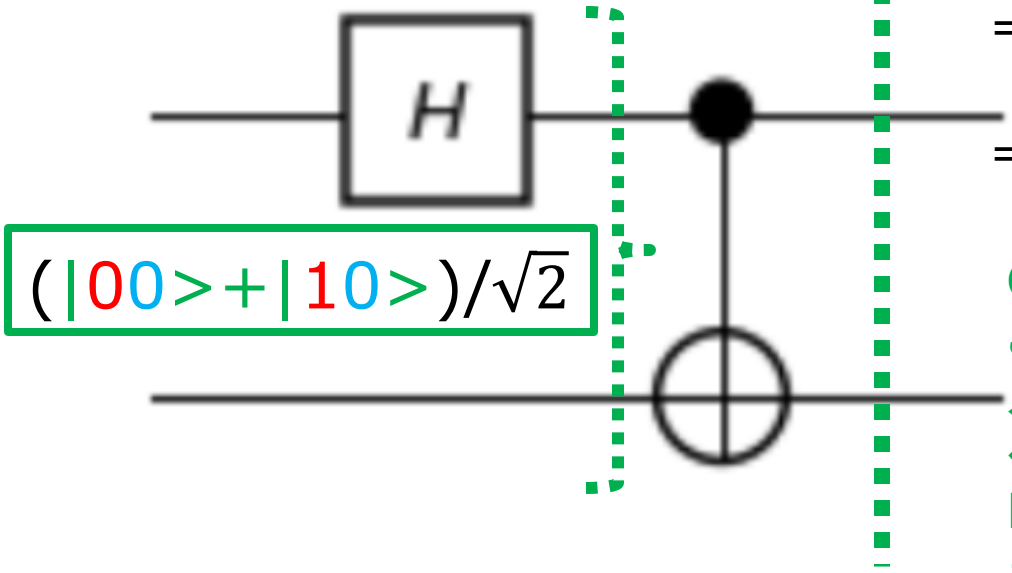
入力|00>の場合



CNOT((|00>+|10>) /√2)
= **CNOT**(|00>) /√2
+**CNOT**(|10>) /√2
= (|00>+|11>) /√2

Bell State ゲートの働き

入力|00>の場合



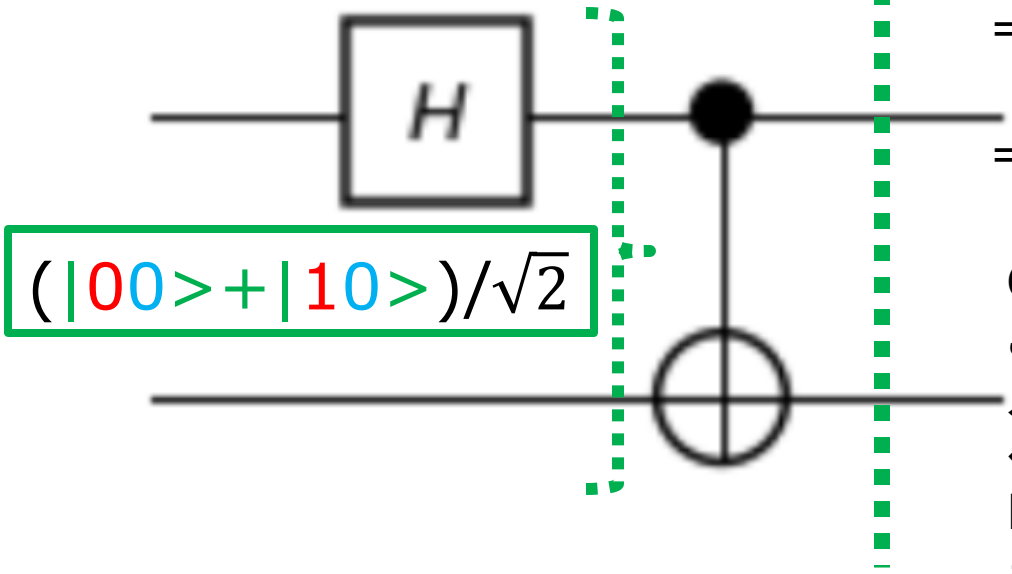
この時点での
系全体の状態
を調べる

CNOT((|00>+|10>) /√2)
= **CNOT**(|00>) /√2
+**CNOT**(|10>) /√2
= (|00>+|11>) /√2

CNOTは線形演算子である。
ここでは、線形演算子Mで
スカラー a,b
ベクトル u,v について
M(au+bv)
= aM(u)+bM(v)
を利用した。

Bell State ゲートの働き

入力|00>の場合

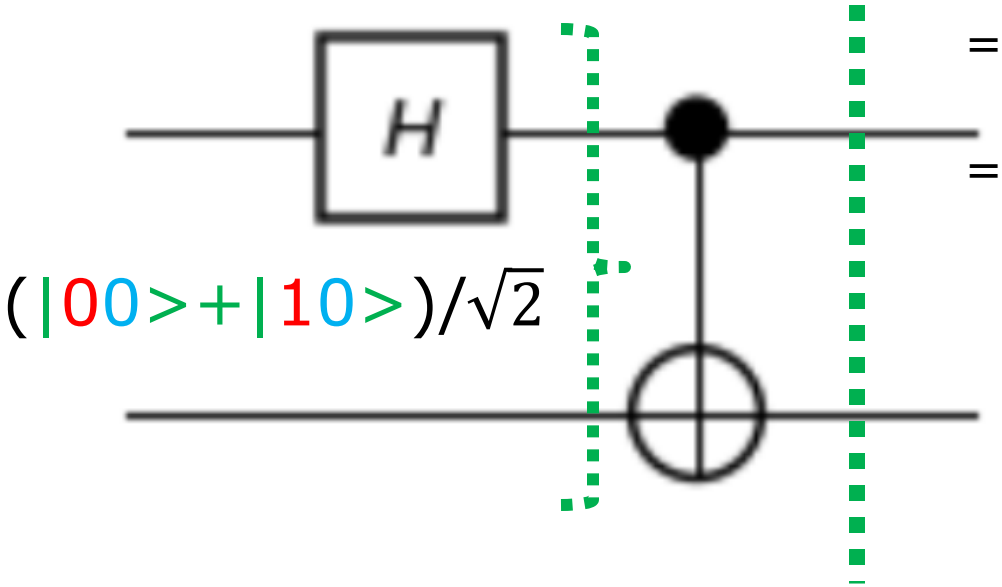


CNOT((|00>+|10>) /√2)
= **CNOT**(|00>) /√2
+**CNOT**(|10>) /√2
= (|00>+|11>) /√2

CNOTは線形演算子である。
ここでは、線形演算子Mで
スカラー a,b
ベクトル u,v について
M(au+bv)
= aM(u)+bM(v)
を利用した。

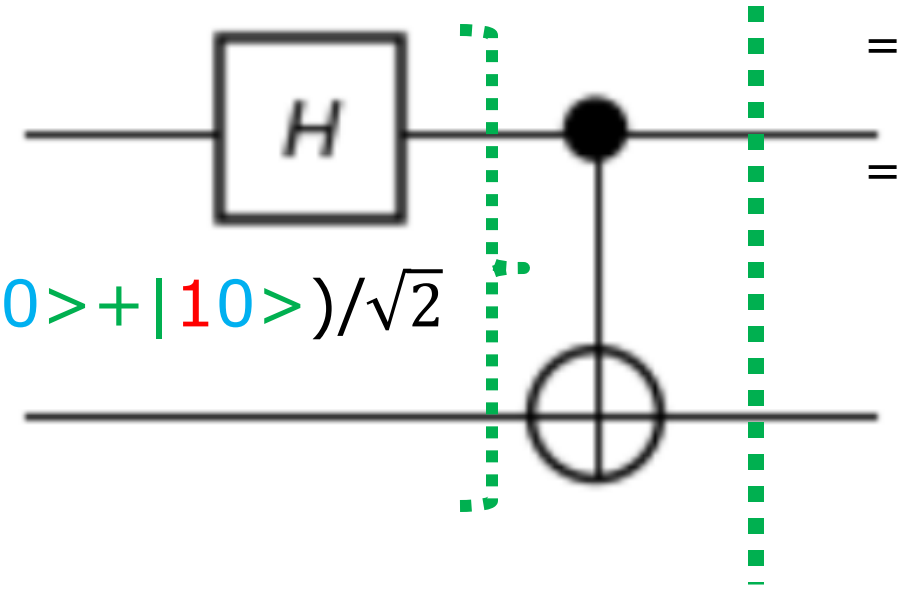
また、
CNOT(|00>) = |00>
CNOT(|10>) = |11>
である。

Bell State ゲートの働き
入力|00>の場合



CNOT((|00>+|10>) /√2)
= **CNOT**(|00>) /√2
+**CNOT**(|10>) /√2
= (|00>+|11>) /√2

Bell State ゲートの働き
入力|00>の場合



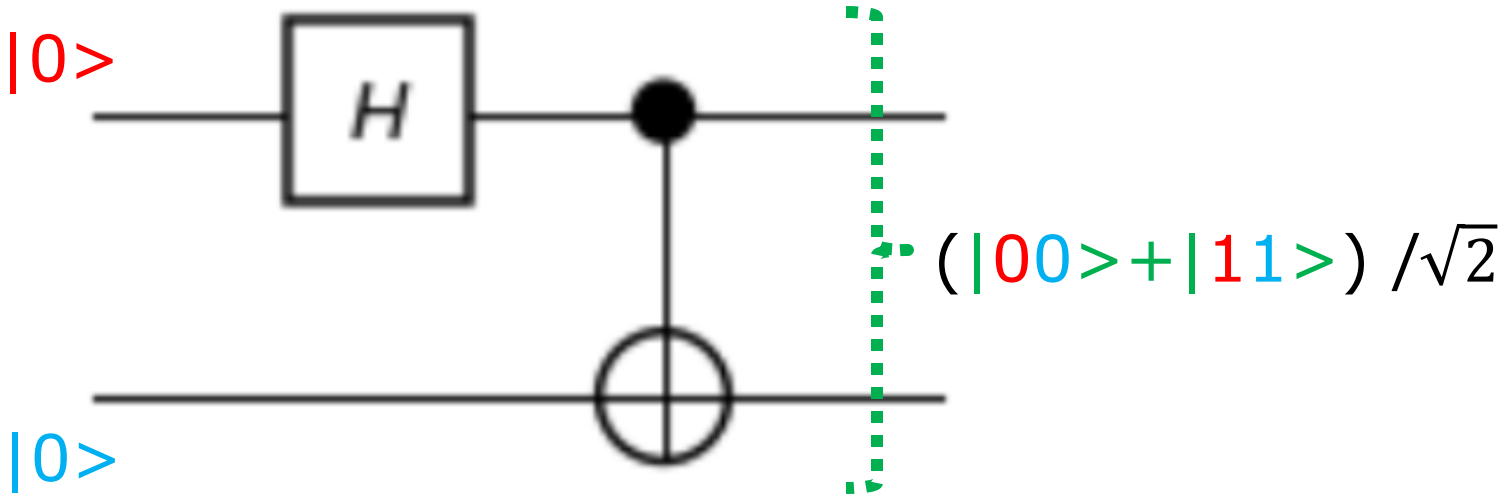
$(|00\rangle+|10\rangle)/\sqrt{2}$

CNOT$((|00\rangle+|10\rangle)/\sqrt{2})$
$=$ **CNOT**$(|00\rangle)/\sqrt{2}$
$+$**CNOT**$(|10\rangle)/\sqrt{2}$
$=(|00\rangle+|11\rangle)/\sqrt{2}$

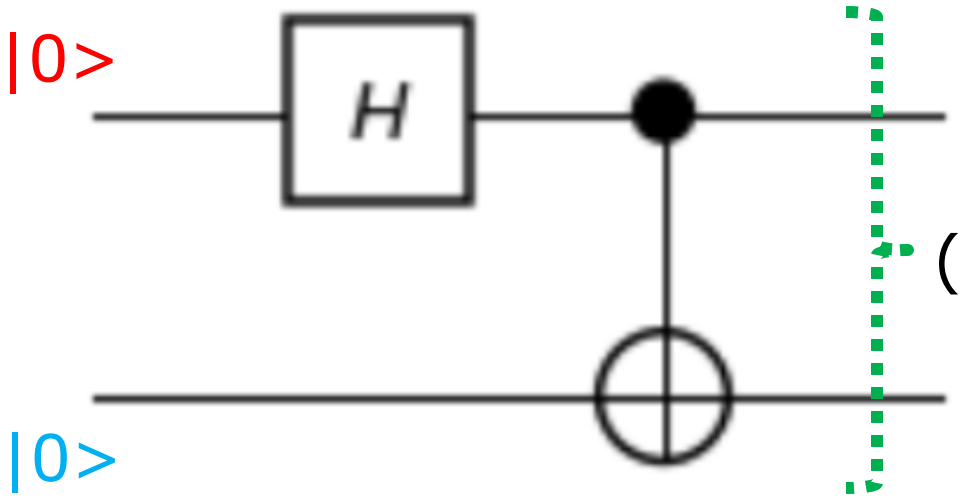
この時点での
系全体の状態

$=(|00\rangle+|11\rangle)/\sqrt{2}$

Bell State ゲートの働き
入力|00>の場合



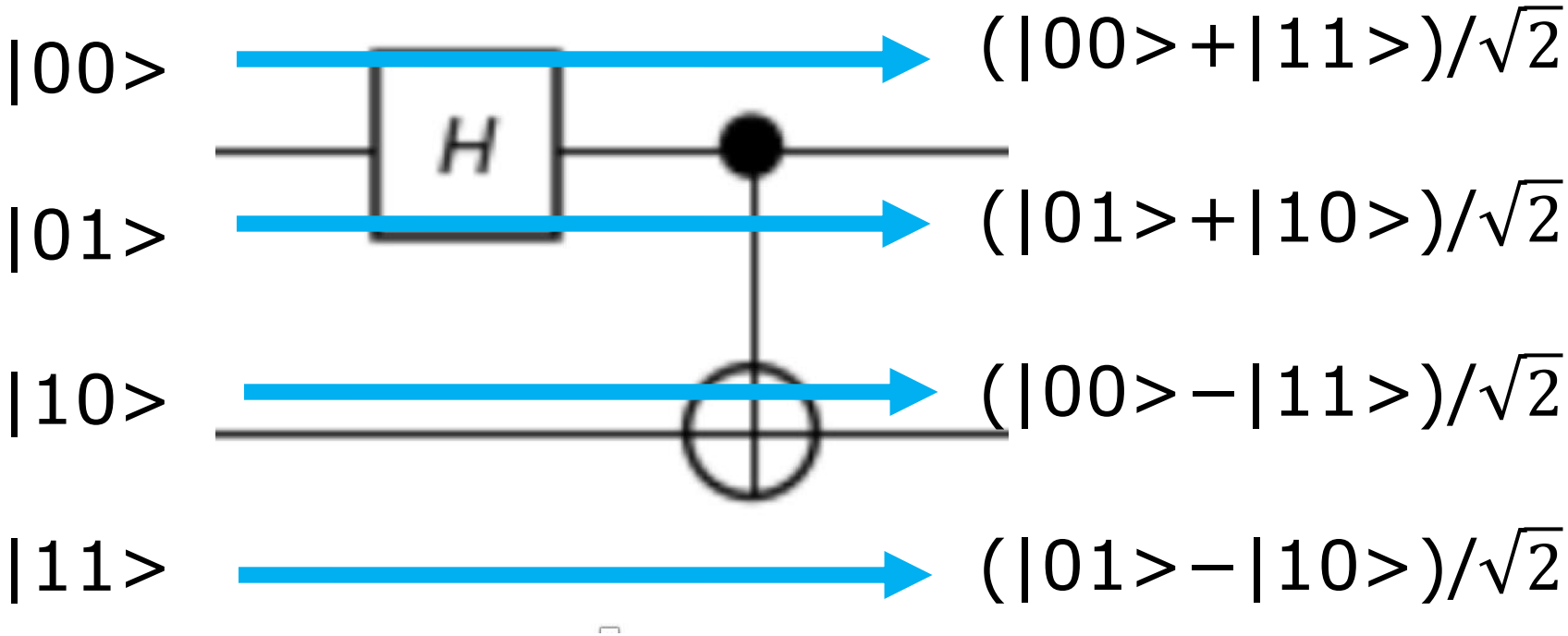
Bell State ゲートの働き
入力|00>の場合



$(|00\rangle+|11\rangle)/\sqrt{2}$

これは、エンタングルメント状態の
Bell Stateの一つ$|\Phi^+\rangle$である。

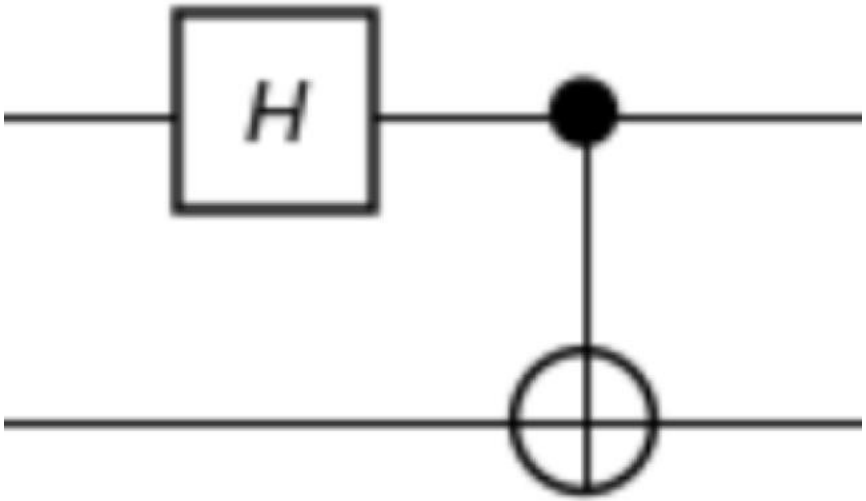
Bell State ゲート



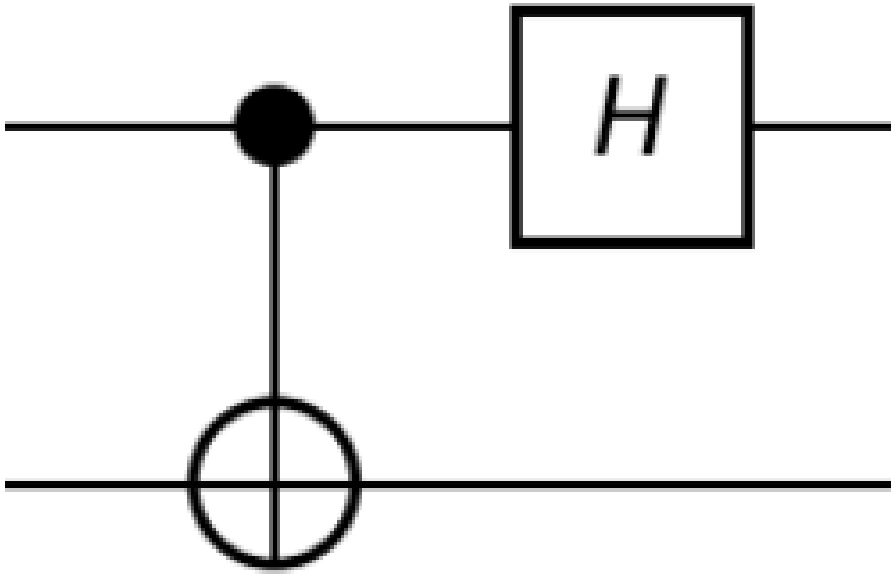
Bell State ゲートの働き

Bell State Gateと
Bell Measure Gate

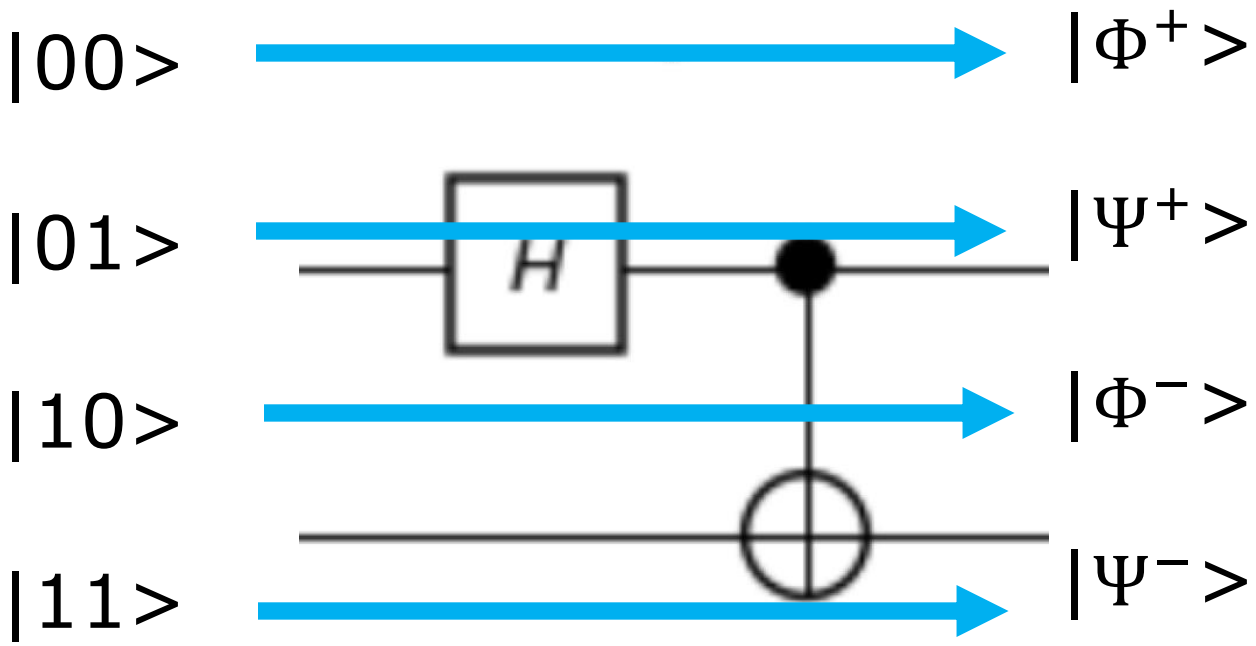
Bell State ゲート



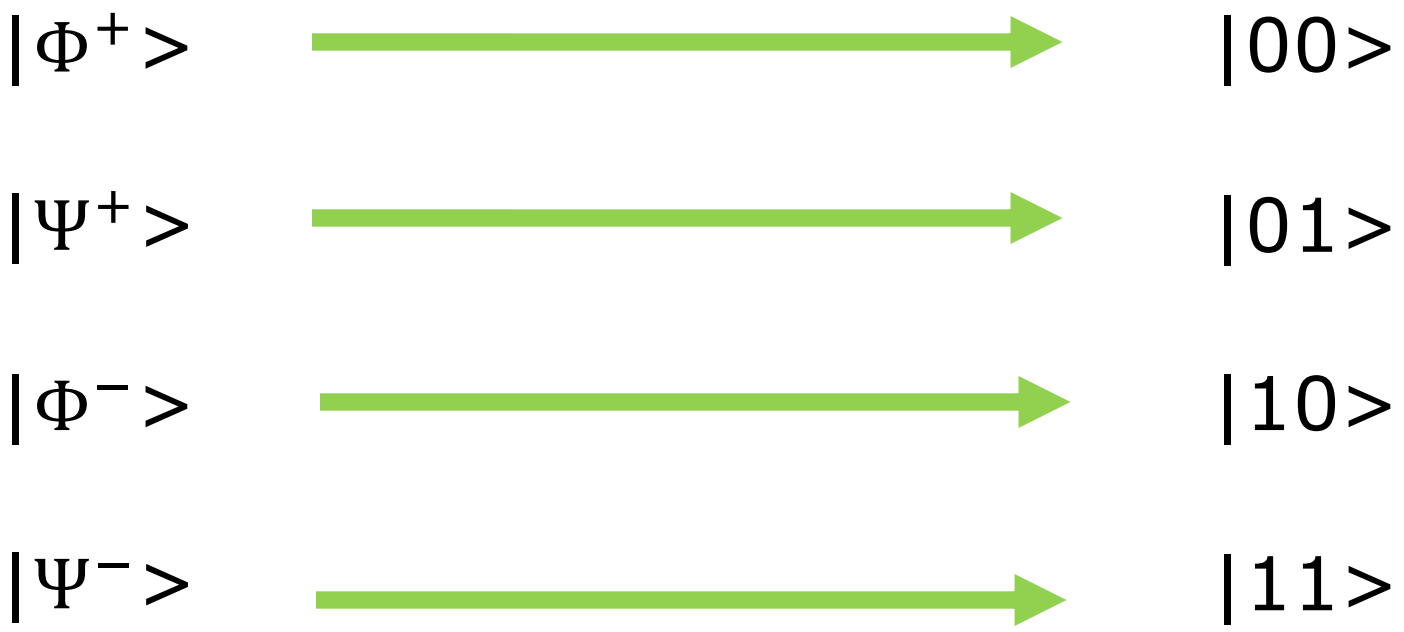
Bell Measure ゲート



Bell State Gate

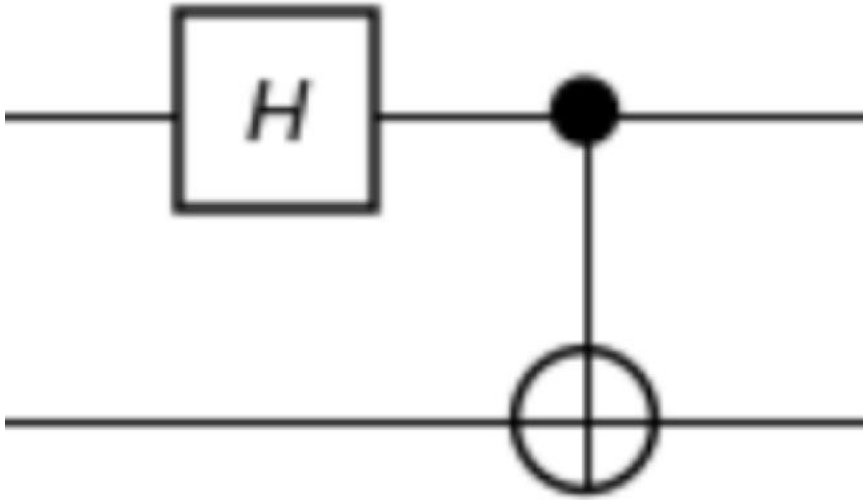


Bell Measure ゲートの働き

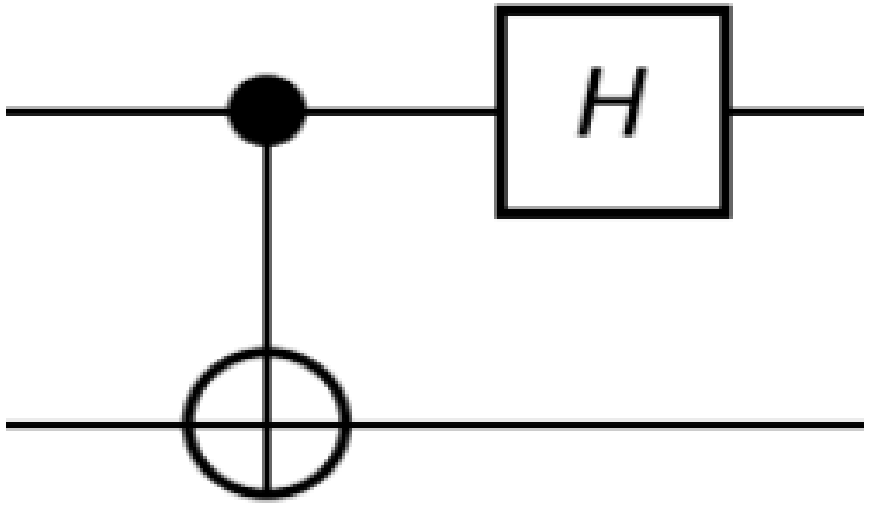


Bell State Gateと
Bell Measure Gate

Bell State Gate

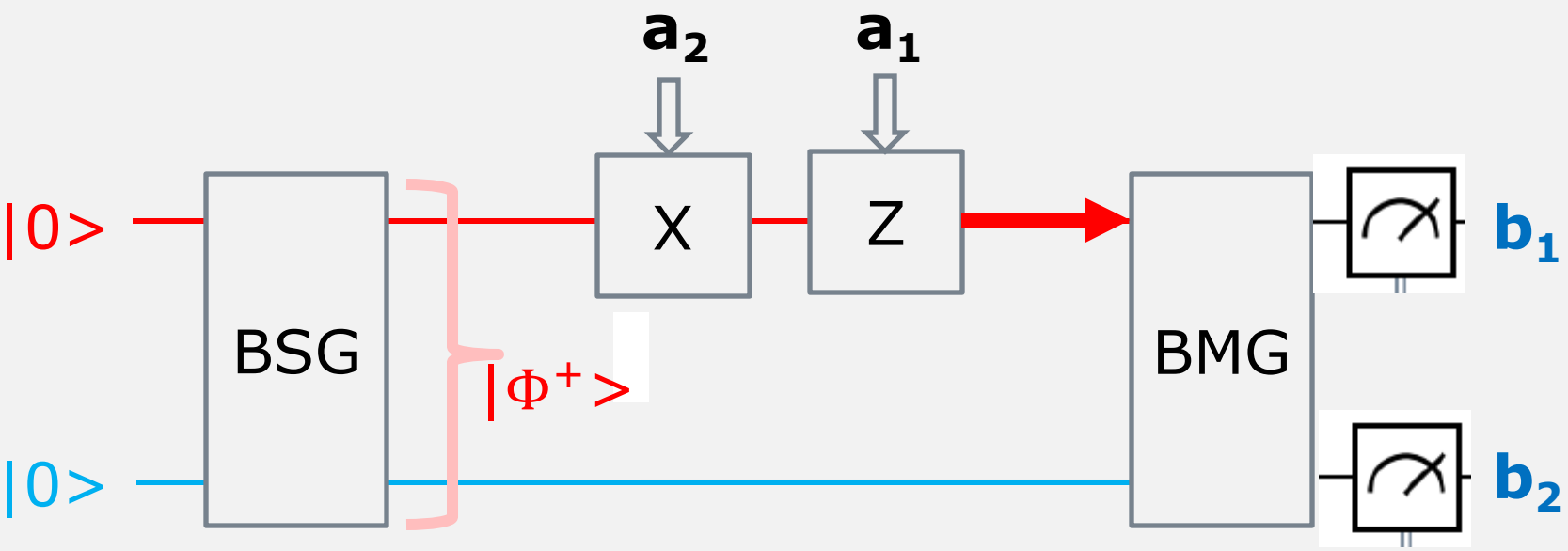


Bell Measure Gate

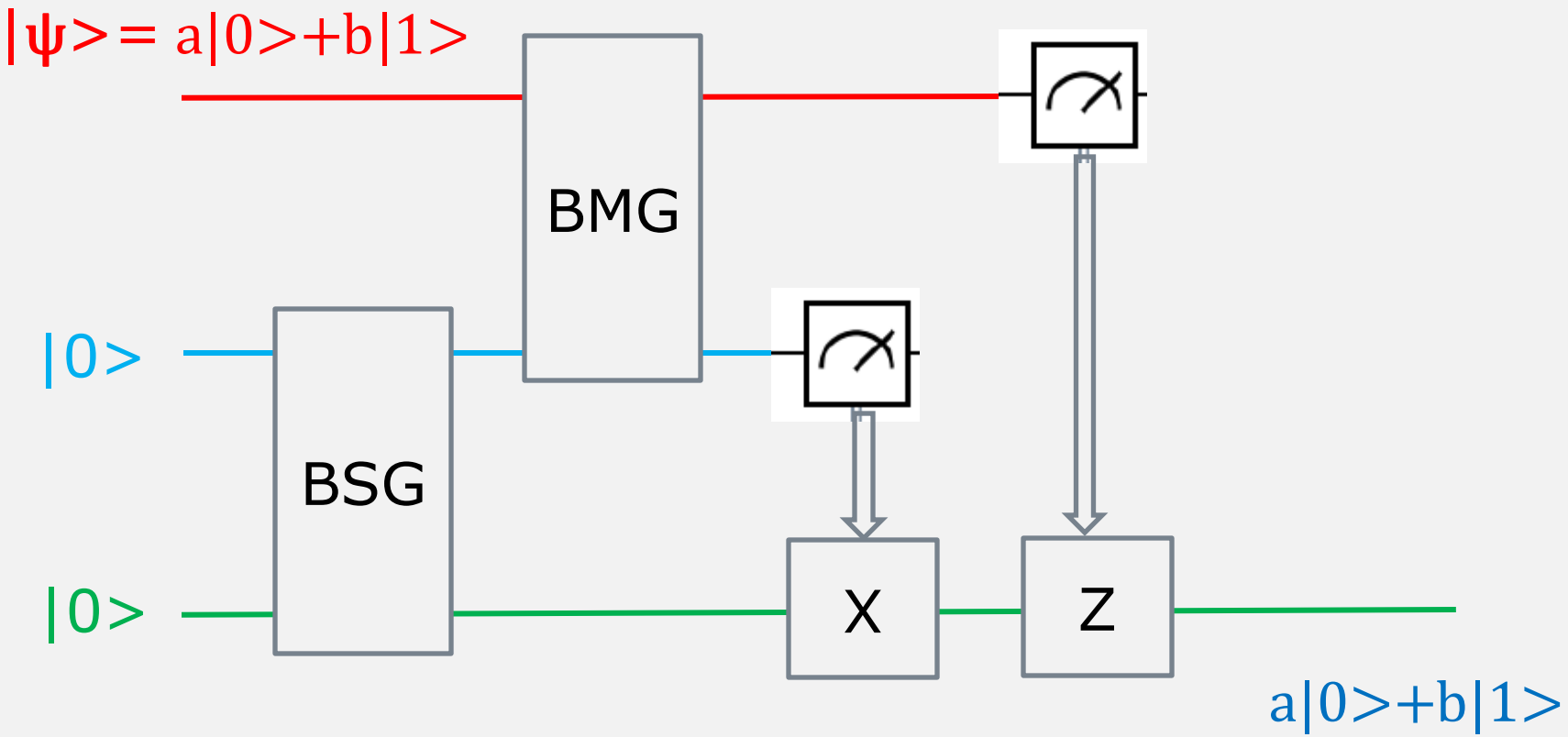


量子通信

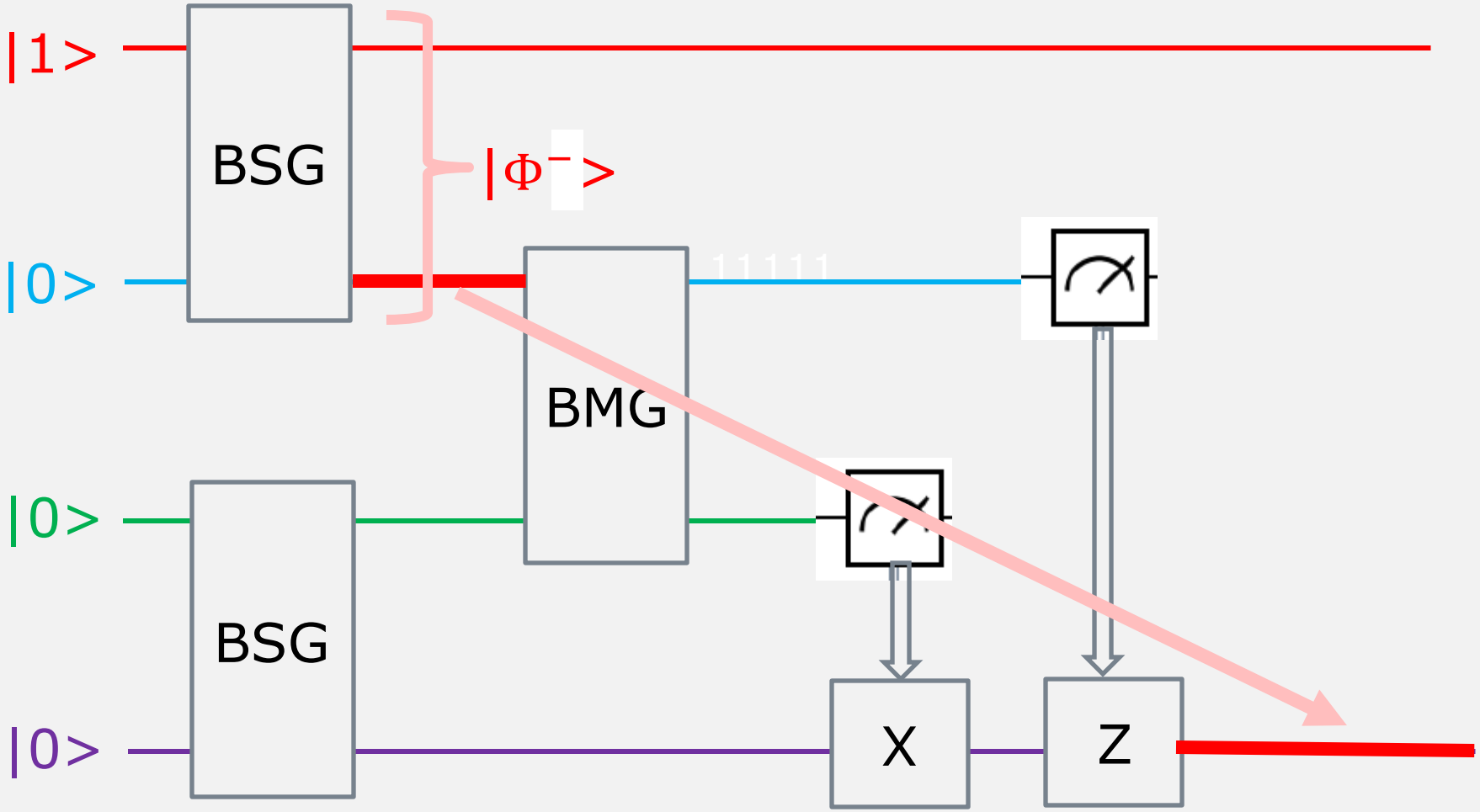
Superdense Coding回路



量子テレポーテーション回路



Entanglement Swapping回路



Part III

量子コンピュータの歴史と到達点

Part III Agenda

量子コンピュータの歴史と到達点

Part III Agenda
量子コンピュータの歴史と到達点

1. ファインマン
   1982年　自然をシミレートする量子コンピュータ
2. ショア
   1994年　量子アルゴリズムの力
3. ローズ (D-Wave)
   2011年　量子コンピュータ実現の困難さと新しい道
4. Google（マルチネス）
   2019年　量子超越性の実証

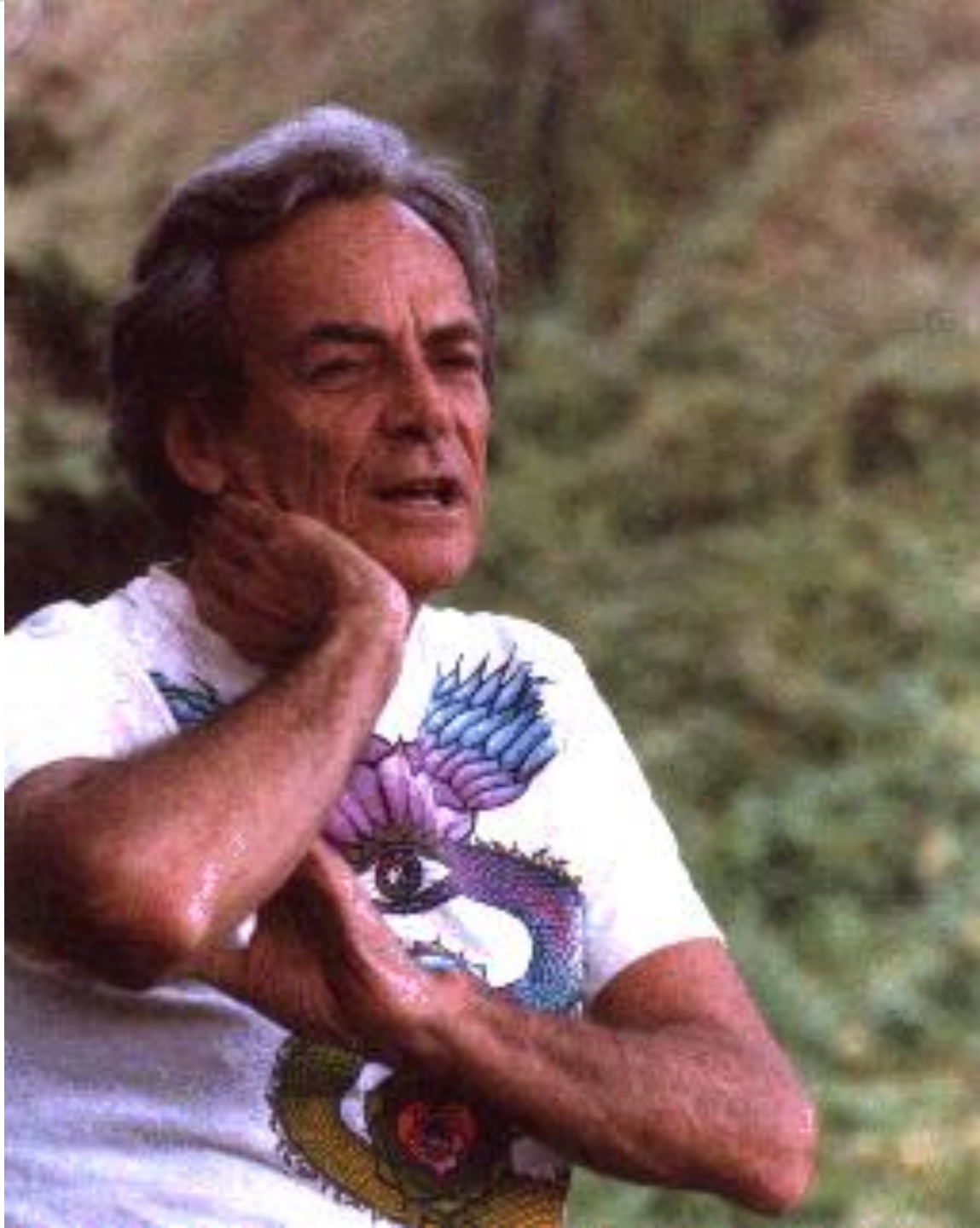
ファインマン

自然をシミレートする量子コンピュータ

1982年

ファインマンの洞察

1982年



Simulating Physics with Computers

1982年 Richard P. Feynman

http://www.cs.berkeley.edu/~christos/classics/Feynman.pdf

自然をシミュレートするコンピュータ

- コンピューターが、正確に自然と同じように振る舞う、正確なシミュレーションが存在する可能性について話そうと思う。
  それが証明されて、そのコンピュータのタイプが先に説明したようなものであるなら、必然的に、有限の大きさの時空の中で起きる全てのものは、有限な数の論理的な操作で正確に分析可能でなければならないことになるだろう。

量子論的システムは、古典的なコンピューターでシミュレートされるか？

- 量子論的なシステムは、古典的な万能計算機で、確率論的にシミュレートされるだろうか？

  別の言い方をすれば、コンピューターは、量子論的なシステムが行うのと、同じ確率を与えるだろうか？　コンピューターを今まで述べてきたような古典的なものだとすれば（前節で述べたような量子論的なものではないとすれば）、また法則はすべて変更されないままで、ごまかしもないとすれば、答えは明らかにノーである。

量子コンピュータ
-- 万能量子シミュレーター

- それは、新しいタイプのコンピューター、量子コンピューター？で可能になるだろう。
  私が理解する限りでは、それは量子論的なシステムによって、量子コンピューターの要素によって、シミュレート出来るようになることは、いまや、明らかになった。それはチューリング・マシンではない。別のタイプのマシンである。

ショア

量子アルゴリズムの力

1994年

ショア

量子コンピュータによる
素因数分解アルゴリズム
の発見

1994年

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer

1995年 Peter W. Shor

http://arxiv.org/pdf/quant-ph/9508027v2.pdf

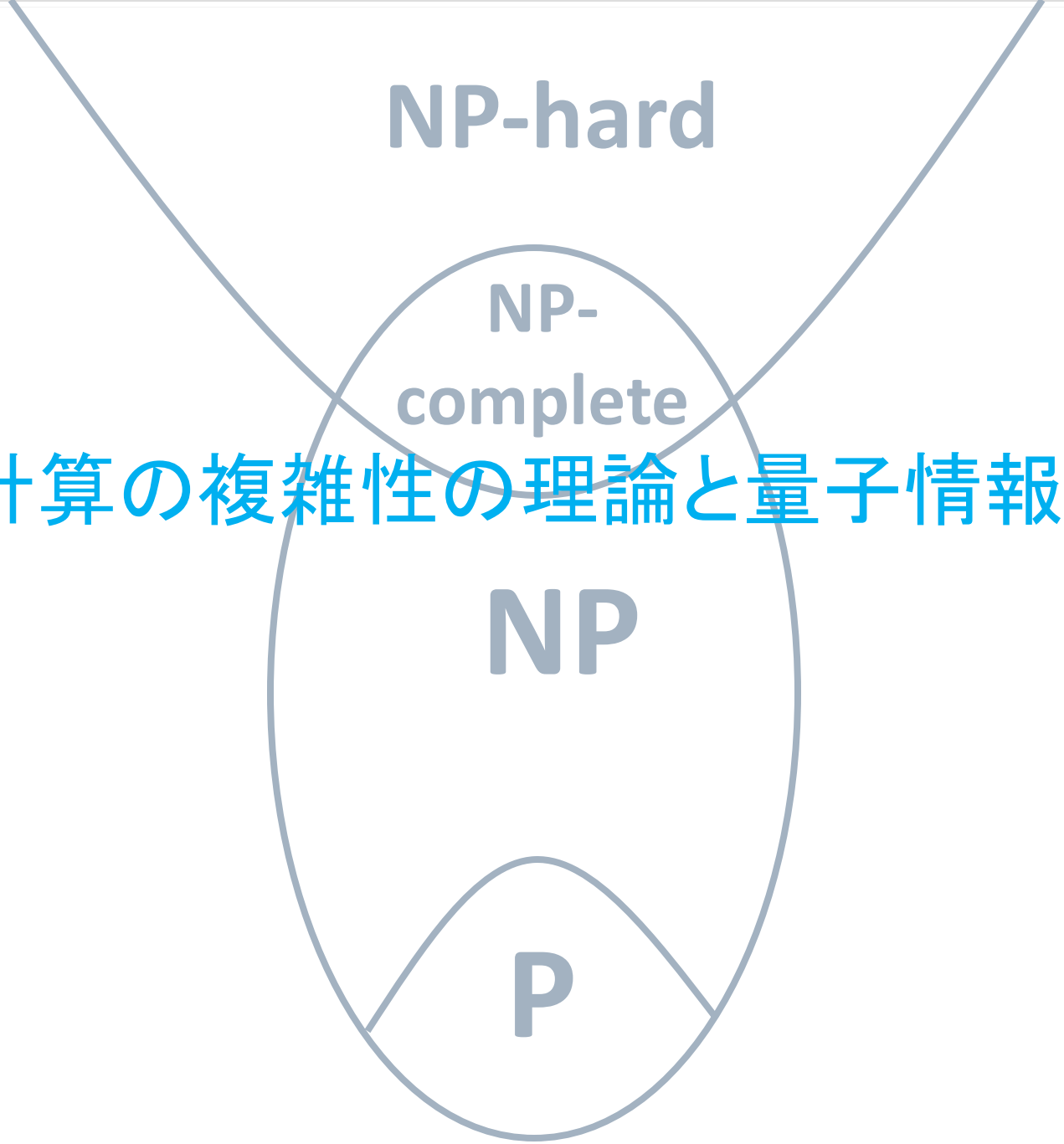
A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

If computers that you build are quantum,
Then spies everywhere will all want 'em.
Our codes will all fail,
And they'll read our email,
Till we get crypto that's quantum, and daunt 'em.

– Jennifer and Peter Shor

To read our E-mail, how mean
of the spies and their quantum machine;
be comforted though,
they do not yet know
how to factorize twelve or fifteen.

– Volker Strassen

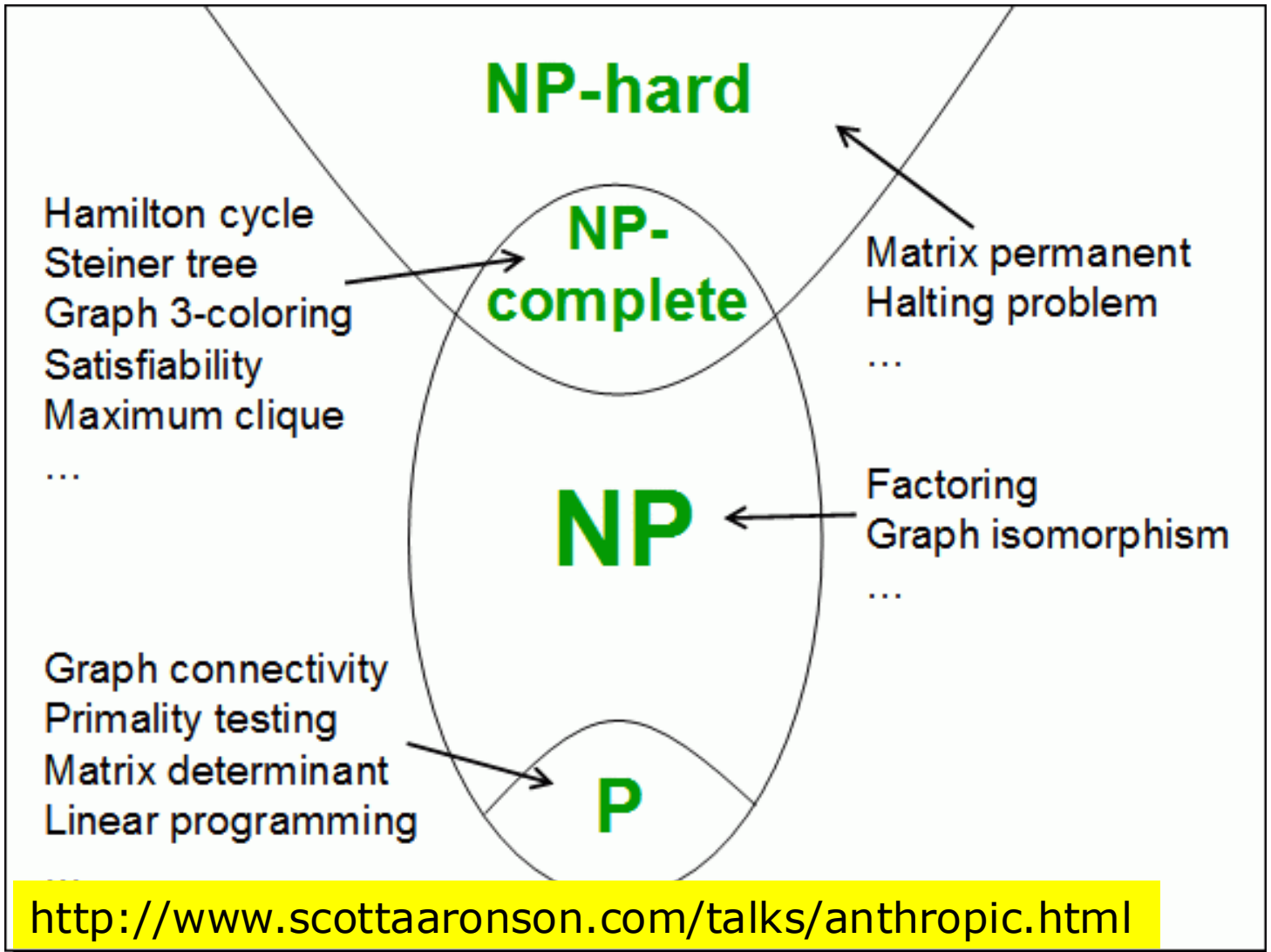


計算の複雑性の理論と量子情報理論

計算量の理論　P≠NP予想

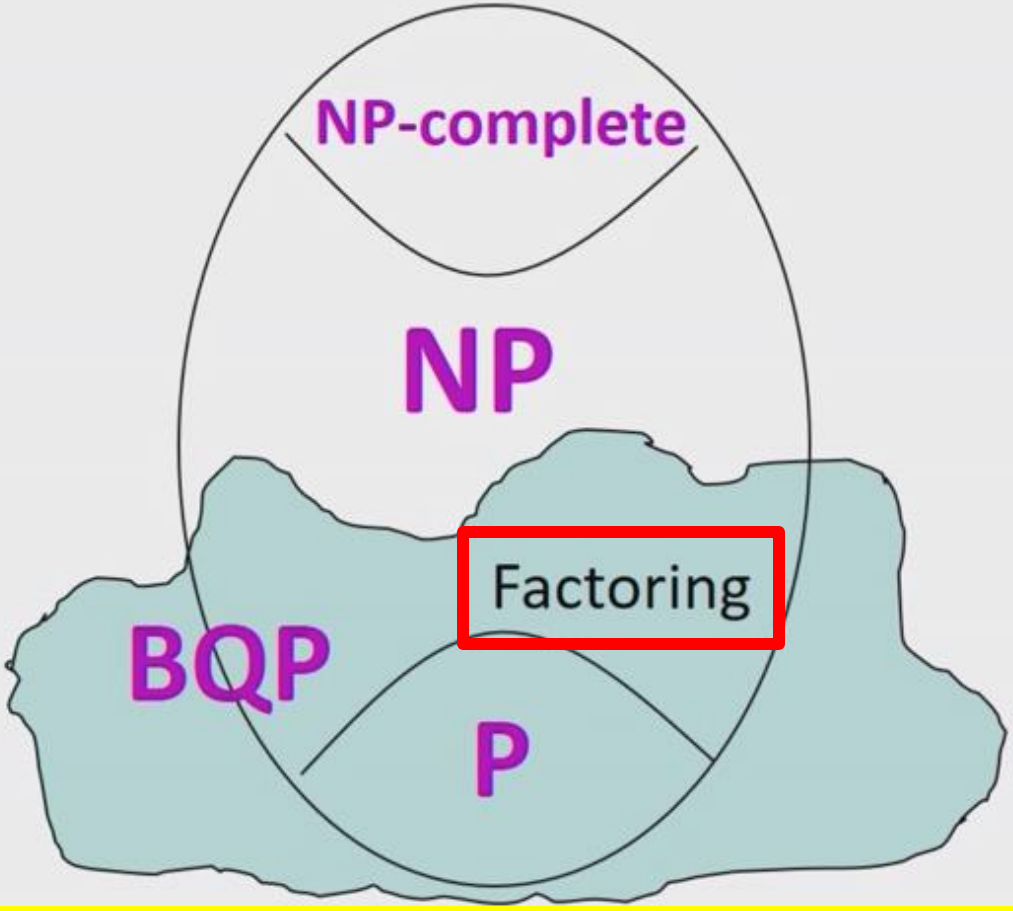
- Pは、多項式時間でTuringマシンで解ける問題。
- NP(Nondeterministic Polynomial)は、その答えがYESなら、そのチェックが多項式時間で解ける問題。
- n＝pqなる素数p,qを求める問題は、NP問題である。nと同時にpあるいはqが与えられた時、それが条件を満たすかのチェックは、すぐに出来るから。
- 例：
  127 x 129 ＝ 29083
  ? x ? ＝ 29083 を満たす？を見つけよ。
  127 x 129 は、29083か？　YES!

計算量の理論　P≠NP予想



http://www.scottaaronson.com/talks/anthropic.html

It Isn't A Magic Bullet That Solves All Problems Instantly



"What Quantum Computing Isn't" Scott Aaronson https://goo.gl/95SpPA

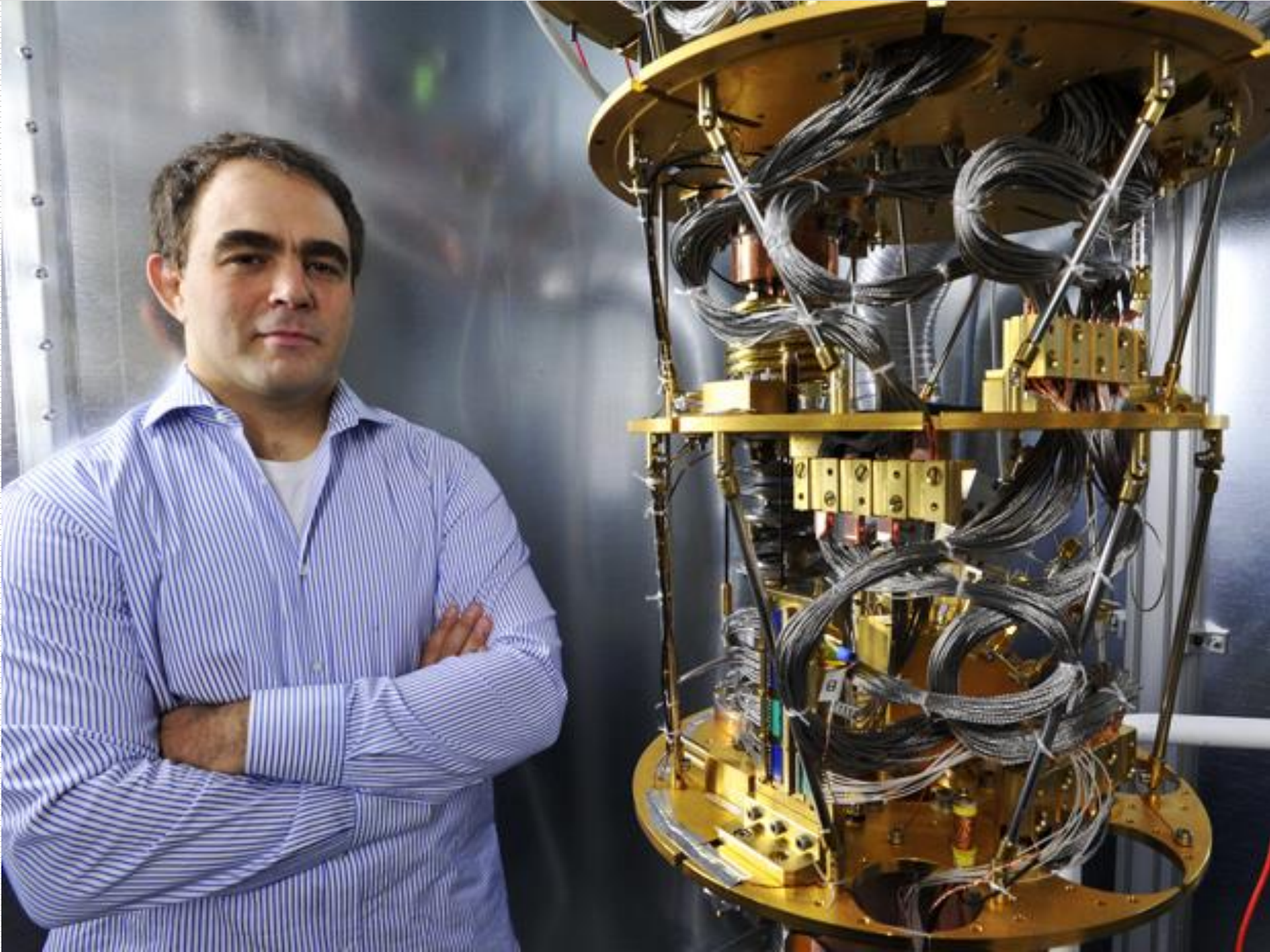
ローズ (D-Wave)

量子コンピュータ実現の困難さと新しい道

2010年

参考資料
「量子コンピューターの新しい潮流 ー D-Waveのアプローチ」
https://www.marulabo.net/docs/20130924-marulec04/

D-Waveマシンと D-Wave社創設者 Geordie Rose



「汎用ゲートモデル」の問題点 by Geordie Rose

- エネルギーの固有状態のsuperpositionを維持して利用する必要がある。その状態は全く「不自然」なもので、極めて破れやすい。
- 「量子エラー訂正」のオーバーヘッドが大きく、高速で高いクォリティのフィードバックを必要とする。
- 必要とされる高速のコントロールは、スケールするのが難しい。
- プロセッサーのアーキテクチャーについての研究は、全く進んでいない。

D-Wave - Natural Quantum Computation (Google Tech talks 2010/10)
http://www.youtube.com/watch?v=56qR0iX5A4o

Roseの述懐

- ゲート・モデルが、どうしたら動くようになるのか、全く見えなかった。2003年には、決断をしなければと思うようになった。選択肢は二つあった。一つは、研究をあきらめること。もう一つの選択は、何か他のものを作ることだった。
- この時、1999年の4月にScience誌に掲載された、Brookeらの”Quantum Annealing of a disordered magnet”という論文に注目した。彼らは、計算を古典的なスピン問題に変換して、それを量子アニーリング効果を使って解くことを提案していた。

Roseの述懐

- 二つの重要なアイデアが生まれた。
- 一つは、古典的なスピン問題を攻略する超伝導プロセッサーを作ること。
- あと一つは、プロセッサーが有用な解を高速で返すように、量子効果を利用することである。

Google（マルチネス）

量子超越性の実証

2019年

参考資料
「量子コンピュータの現在 ー 量子優越性のマイルストーンの達成 ー」
https://www.marulabo.net/docs/q-supremacy/

Quantum supremacy using a programmable superconducting processor

2019/10/23

https://www.nature.com/articles/s41586-019-1666-5

Google 論文は、多くの著者の連名で書かれている

Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, **Edward Farhi**, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J.Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven & **John M. Martinis**

論文の概要

量子コンピューターが約束していることは、特定の計算タスクが古典プロセッサーよりも量子プロセッサー上では、指数関数的に高速に実行される可能性があることである。

基本的な挑戦は、指数関数的に大きな計算空間上で量子アルゴリズムを実行できる高い信頼性を持つプロセッサを構築することである。

この論文では、プログラム可能な超伝導量子ビットを備えたプロセッサを使用して、53量子ビットの量子状態を作成したことを報告する。この量子状態は、次元2^{53}（約10^{16}）の計算状態空間に対応する。

繰り返される実験からの測定は、結果の確率分布をサンプリングすることで行われる。この結果は、古典的なシミュレーションを使用して検証する。

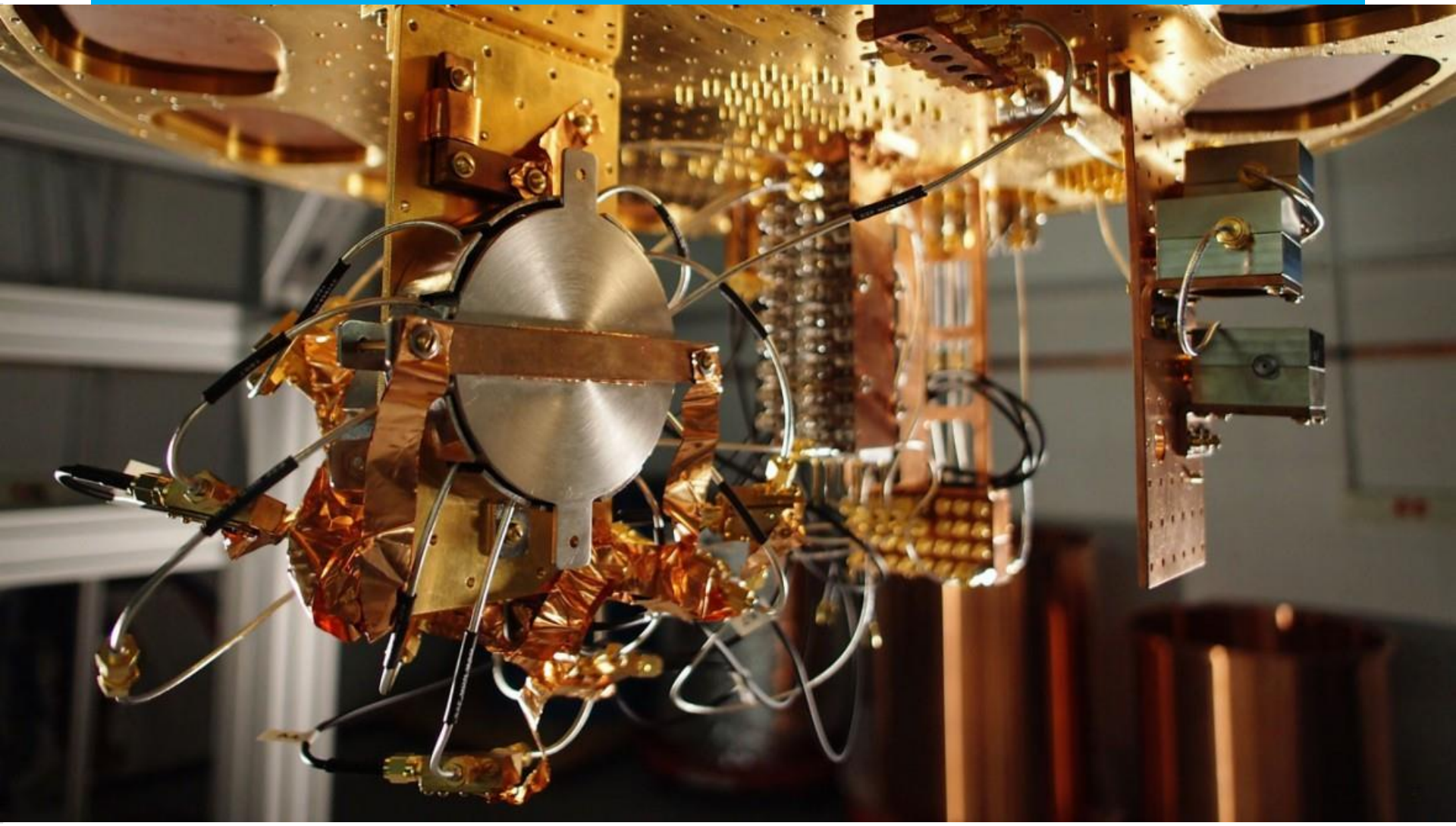
論文の概要

我々のSycamoreプロセッサでは、量子回路の1つのインスタンスを100万回サンプリングするのに約200秒を要した。

現在のベンチマークでは、最先端の古典的なスーパーコンピューターで同等のタスクを実行するには、約10,000年かかる。

すべての既知の古典的なアルゴリズムと比較して、この劇的な速度の向上は、この特定の計算タスクに対して量子優位性を実験的に実現したものとなる。この実験は、待望のコンピューティングパラダイムの先駆けである。

Google’s Quantum Dream Machine



https://goo.gl/Y9zvV5

Google's Quantum Dream Machine
John Martinis

- The theoretical underpinnings of quantum computing are well established. And physicists can build the basic units, known as qubits, out of which a quantum computer would be made. They can even operate qubits together in small groups. But they have not made a fully working, practical quantum computer.
- Martinis is a towering figure in the field: his research group at the University of California, Santa Barbara, has demonstrated some of the most reliable qubits around and gotten them running some of the code a quantum computer would need to function.

https://goo.gl/MTrLFX

John Martinis

- He was hired by Google in June 2014 after persuading the company that his team's technology could mature rapidly with the right support. With his new Google lab up and running, Martinis guesses that he can demonstrate a small but useful quantum computer in two or three years. "We often say to each other that we're in the process of giving birth to the quantum computer industry," he says.

https://goo.gl/MTrLFX

Part IV

量子コンピュータと人工知能論

Part IV Agenda
量子コンピュータと人工知能論

1. 「機械は考えることができるか？」
2. 現代の生命観 -- 分子機械としての人間
3. 機械の知能の計算主義モデル
4. 複雑性理論の知見
5. 人間と機械の「共生」について

参考資料
「人工知能と計算科学」
https://www.marulabo.net/docs/aicomp/

「機械は考えることができるか？」

「機械は考えることができるか？」

今から約70年前、「機械は考えることができるか？」と問いかける論文がでる。今日の人工知能研究の出発点と言っていいチューリングの論文である。

この論文には、とても印象的な一節がある。

「「機械は考える事が出来るか？」という、最初に掲げた問題が、今では議論にも値しない程無意味なものである事は私も認めよう。… しかし、それにもかかわらず、今世紀の終わりには人々の意見が大きく変化して、ついには、矛盾していると考えることなく『機械の恩考』について語ることができるようになるであろうと私は信じている。」

「機械は考えることができるか？」

チューリングの予想は、70年の時を超えて、見事に的中した。
今では、「人工知能」「AI」という言葉を、メディアで聴かない日はないと言っていいくらいである。僕は、皆が、「矛盾していると考えることなく『機械の恩考』について語」り始めている、この変化を歓迎している。それは、いいいことだ。

ただ、忘れてはならないのは、チューリングは、自身の「機械的知性」の議論を「異端」と感じていたということである。今日では皆の常識になっている「原子論」も、19世紀においては、ボルツマンの学説は「異端」視されていた。

チューリングの予想が的中し、「人々の意見が大きく変化」した今、「機械の思考」について、また、「機械の思考」の到達可能な範囲について、改めて考える条件は拡大していると僕は考えている。

機械的知性　ある異端的理論

INTELLIGENT MACHINERY, A HERETICAL THEORY

"You cannot make a machine to think for you". This is a commonplace that is usually accepted without question. It will be the purpose of this paper to question it.

Most machinery developed for commercial purposes is intended to carry out some very specific job, and to carry it out with certainty and considerable speed. Very often it does the same series of operations over and over again without any variety. This fact about the actual machinery available is a powerful argument to many in favour of the slogan quoted above. To a mathematical logician this argument is not available, for it has been shown that there are machines theoretically possible which will do something very close to thinking. They will, for instance, test the validity

http://www.turingarchive.org/viewer/?id=458&title=1

現代の生命観
-- 分子機械としての人間

現代の生命観
-- 分子機械としての人間

「機械の思考は可能か？」というTuringの問題意識にとって、念頭にあったのは、デカルトの時代から存在していた「人間機械論」だったと僕は考えている。彼の「機械的思考」についての深い「異端」感は、そこにあったと感じている。

ワトソン＝クリックによるDNAの構造の発見は、1953年だが、Turingは1954年に亡くなっている。今日、多くの人が共有している「人間＝分子機械論」は、Turingの生きた時代では「異端」だった。

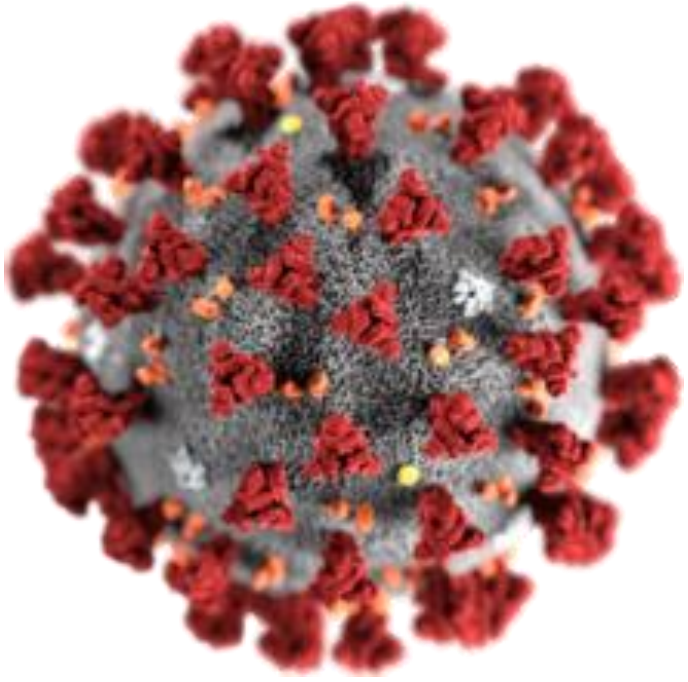
「機械の思考は可能か？」という問題意識にとって、人間も機械的なものであるという認識に至るのは「必然的」なものであったと僕は考えている。

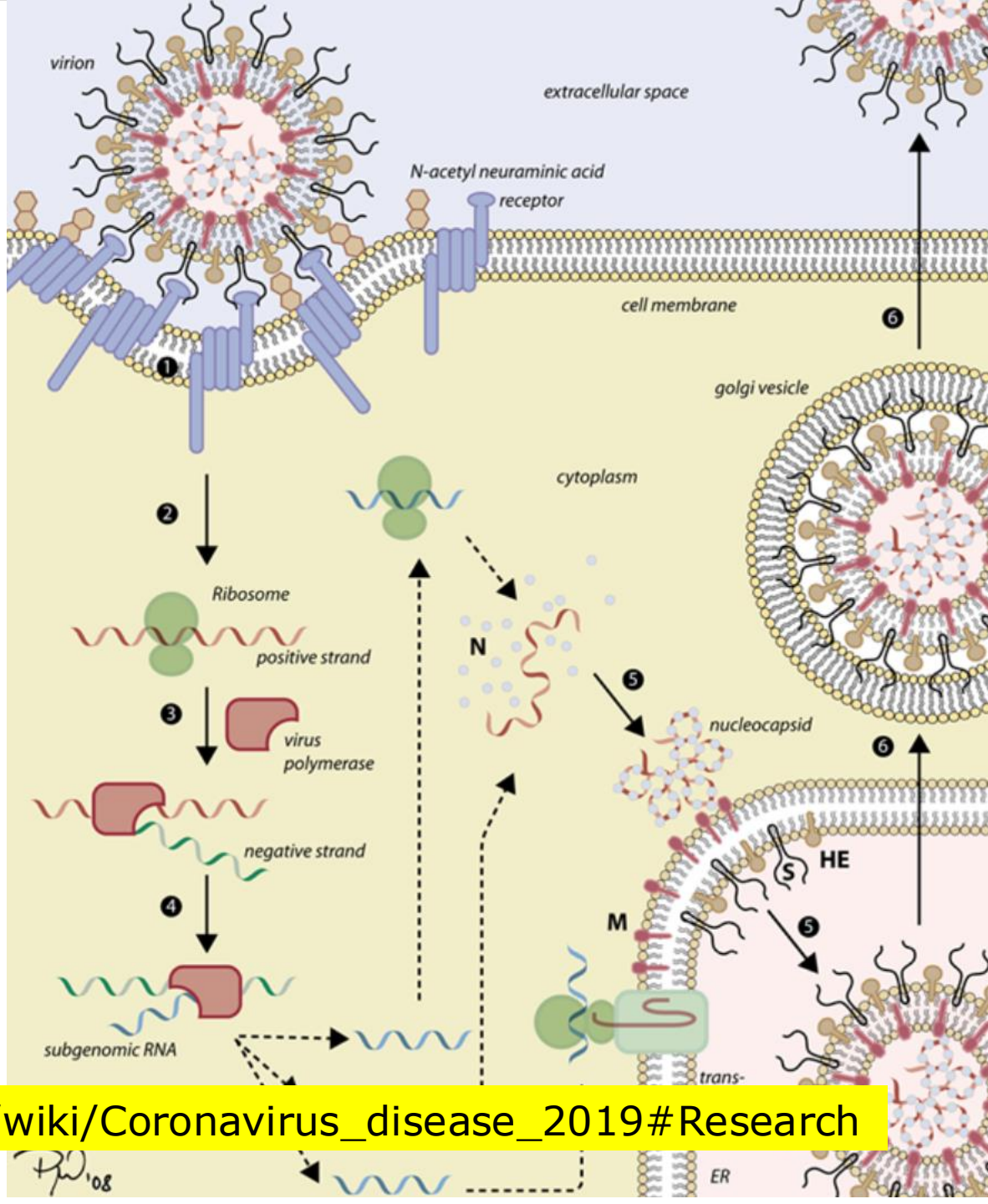
過去のパンデミックとは、疫病の原因とそれへの対応について、明らかに異なる認識が生まれている。それは、「分子機械としての人間」という認識を、さらに広げるものだ。

17世紀　黒死病

20世紀　スペイン風邪

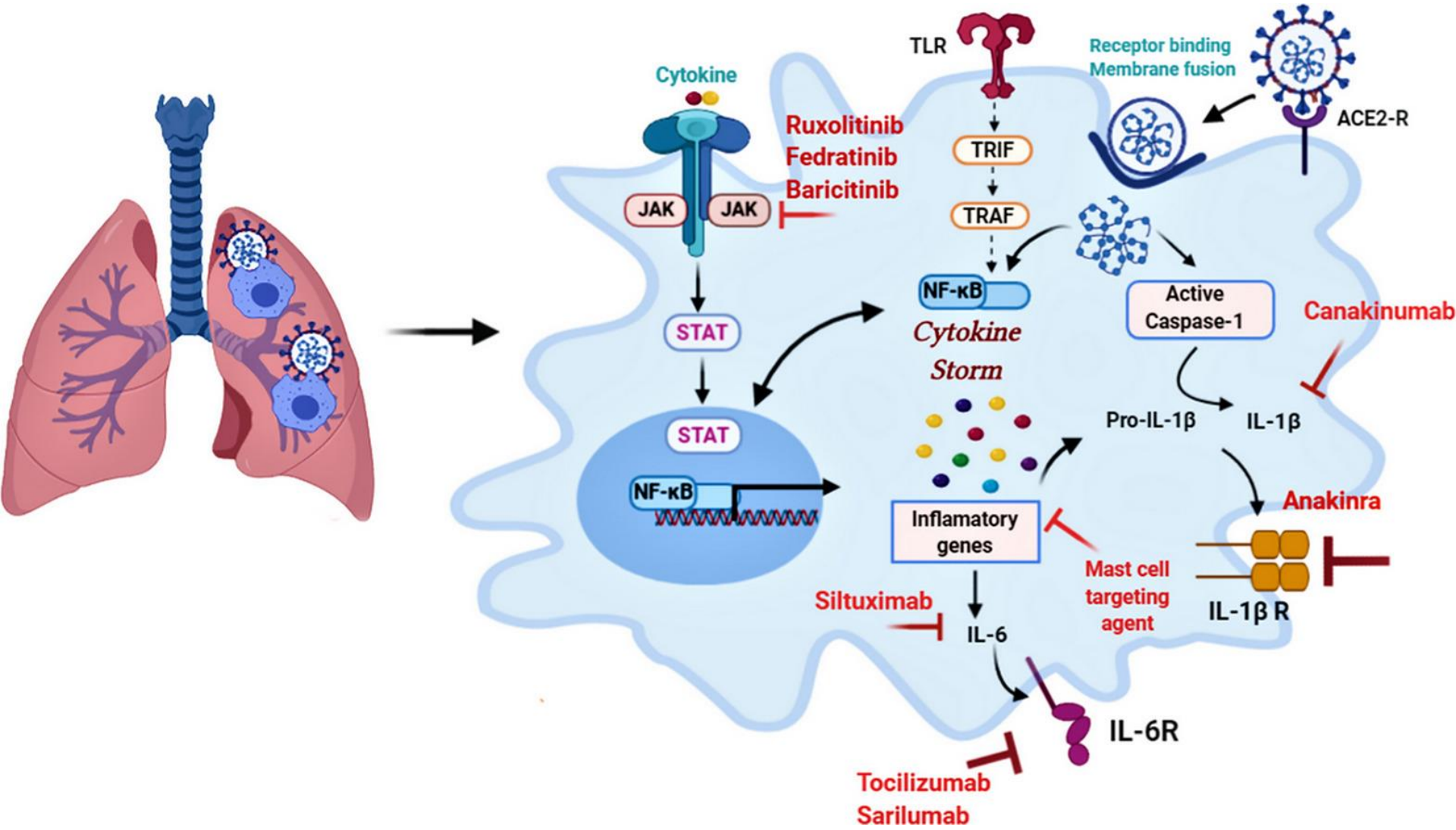
Covid-19



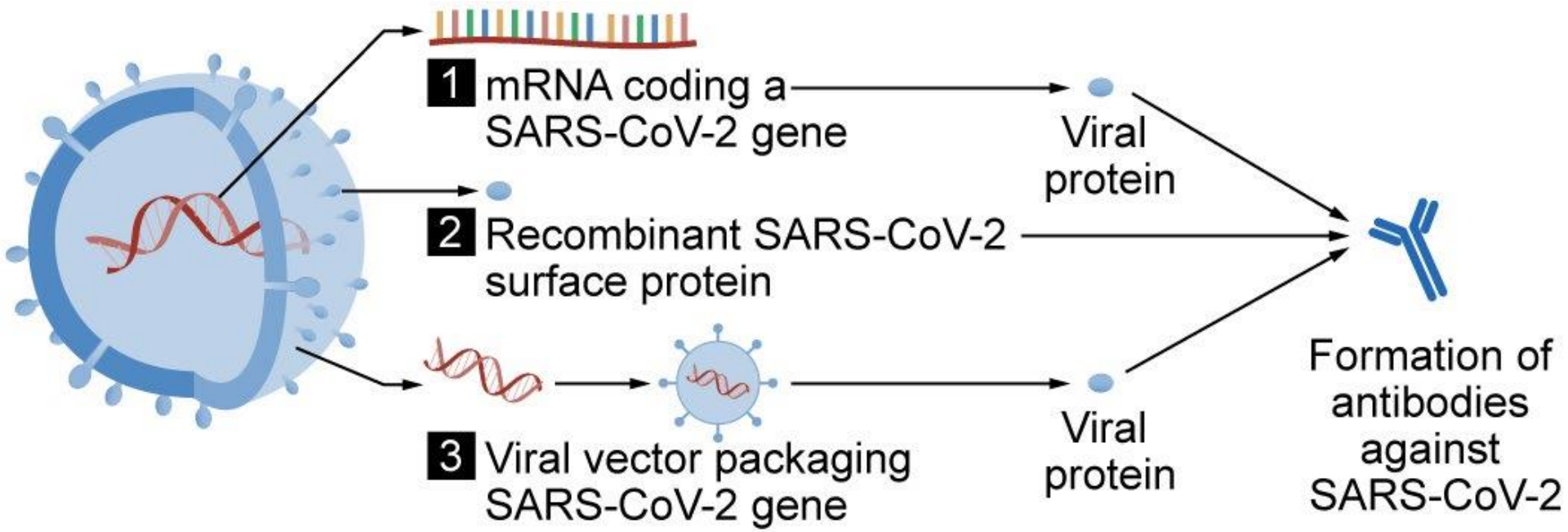


https://en.wikipedia.org/wiki/Coronavirus_disease_2019#Research

Covid-19 Cytokine storm



Covid-19 vaccine

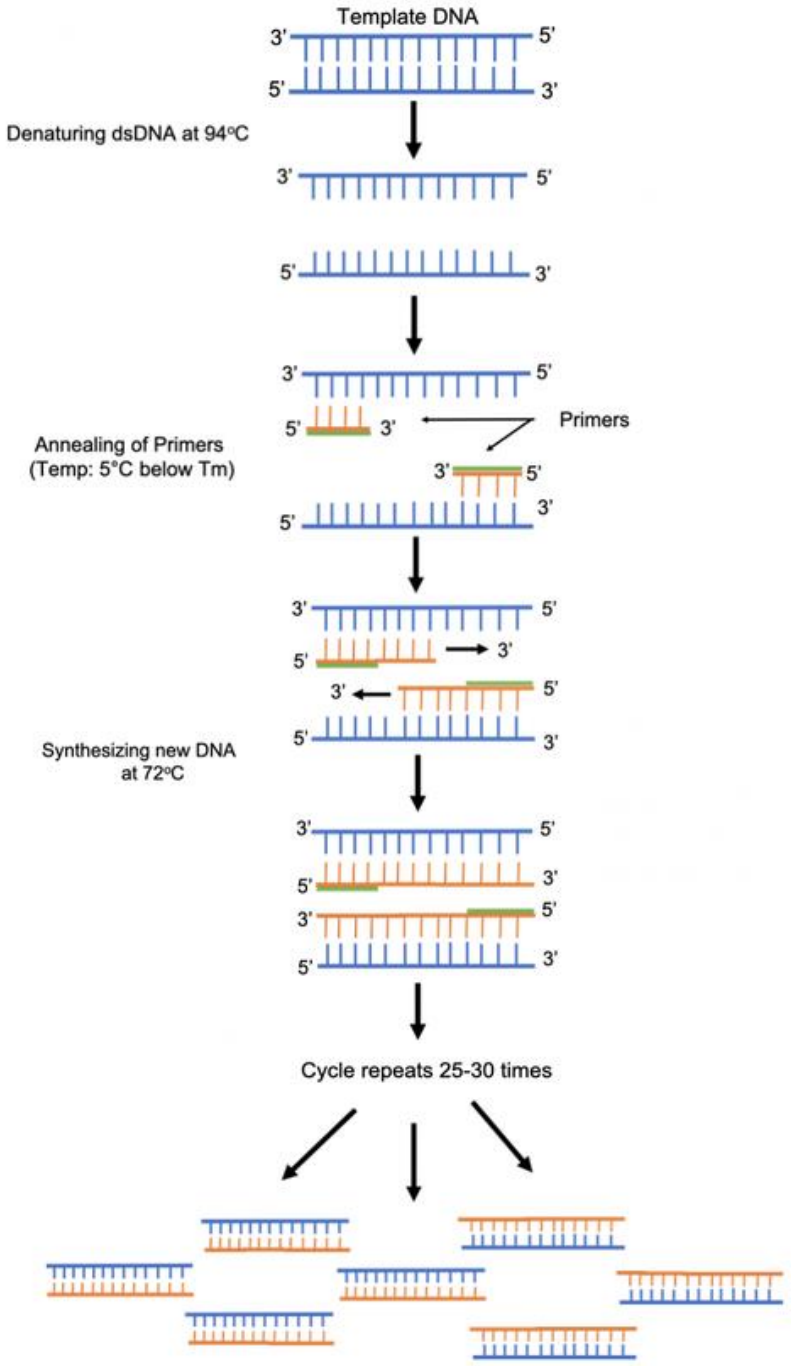


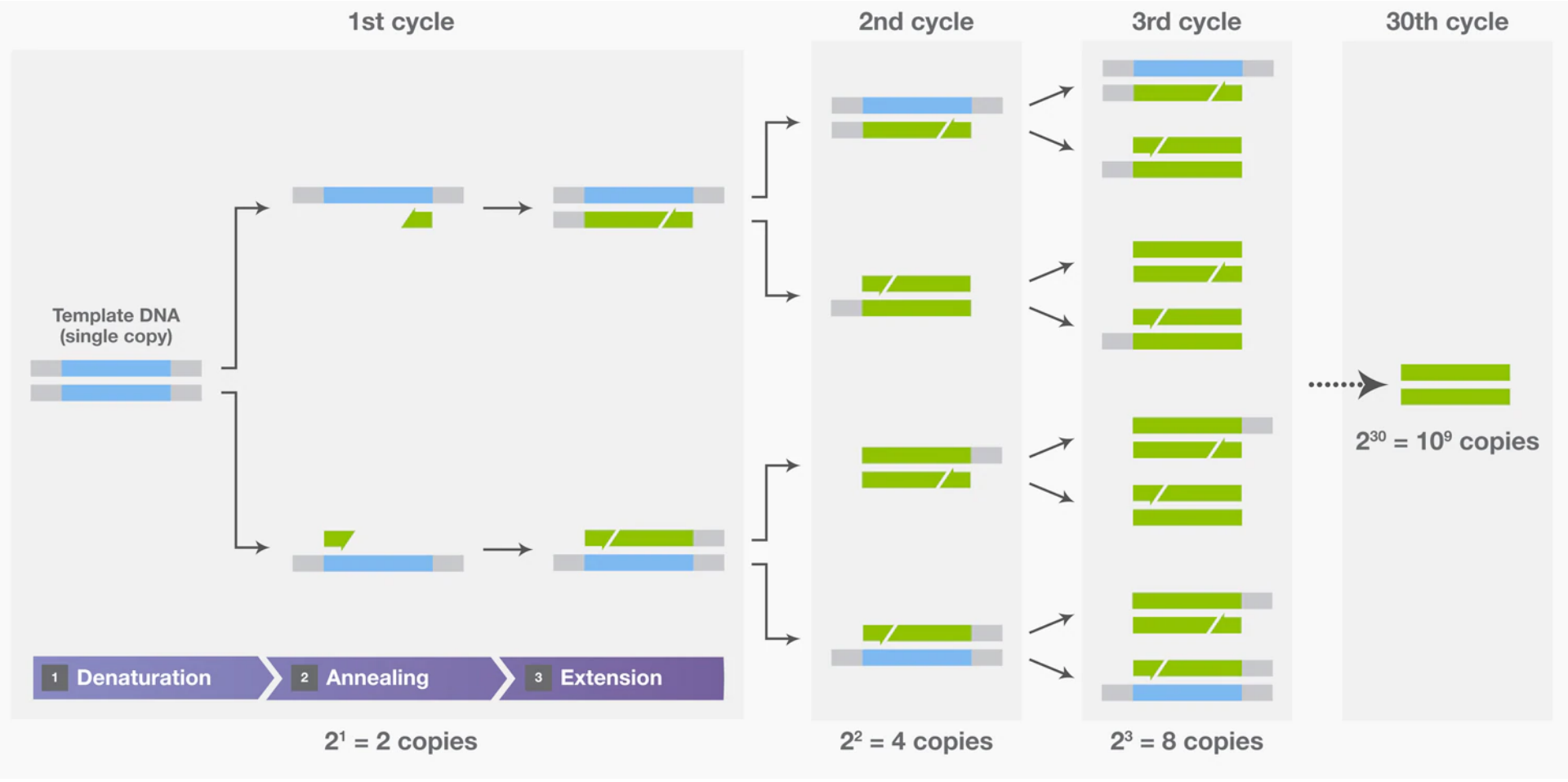
Source: GAO. | GAO-20-583SP

https://en.wikipedia.org/wiki/COVID-19_vaccine

PCR

- Polymerase chain reaction (PCR) is a method widely used to rapidly make millions to billions of copies of a specific DNA sample, allowing scientists to take a very small sample of DNA and amplify it to a large enough amount to study in detail. PCR was invented in 1984 by the American biochemist Kary Mullis at Cetus Corporation.
- 1993年　ノーベル化学賞

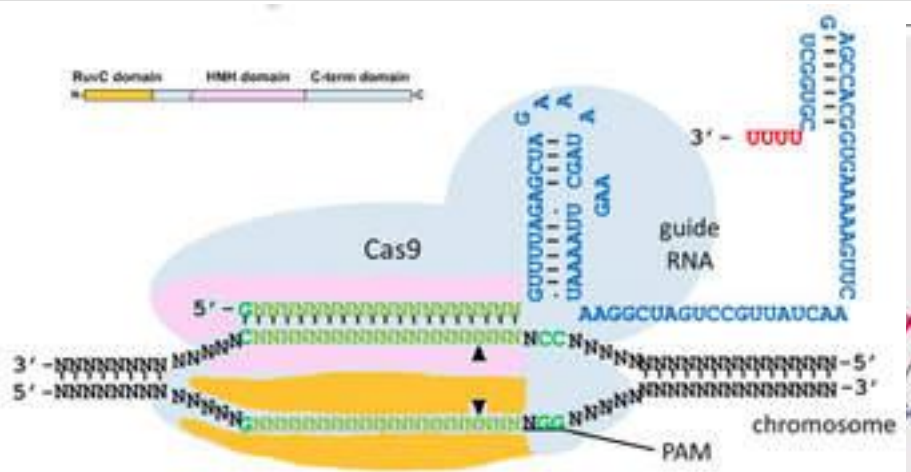


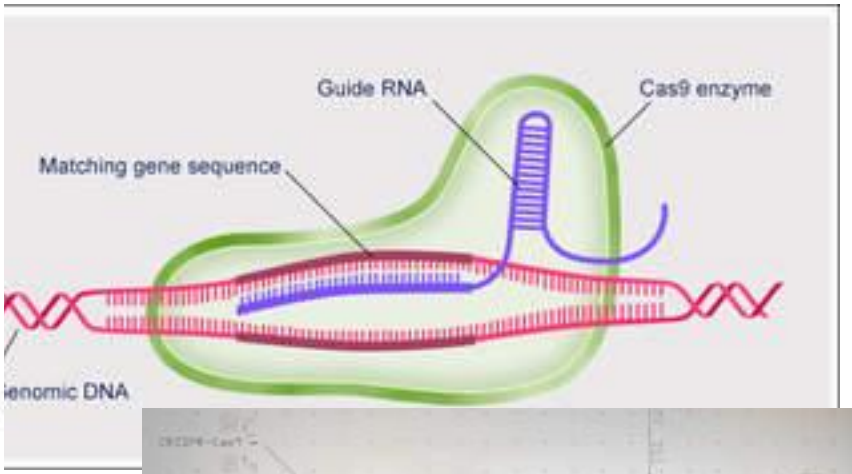

1st cycle
2nd cycle
3rd cycle
30th cycle
Template DNA
(single copy)
1 Denaturation
2 Annealing
3 Extension
2^1 = 2 copies
2^2 = 4 copies
2^3 = 8 copies
$2^{30} = 10^9$ copies

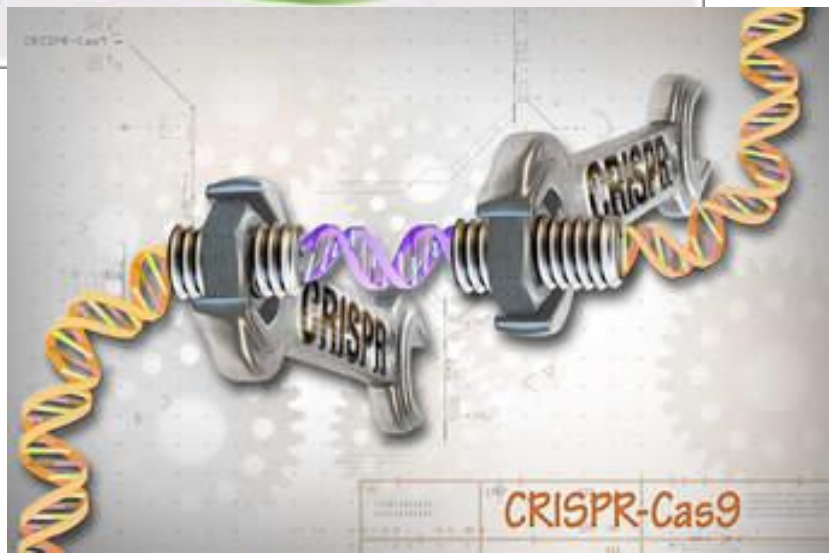
2020 ノーベル科学賞

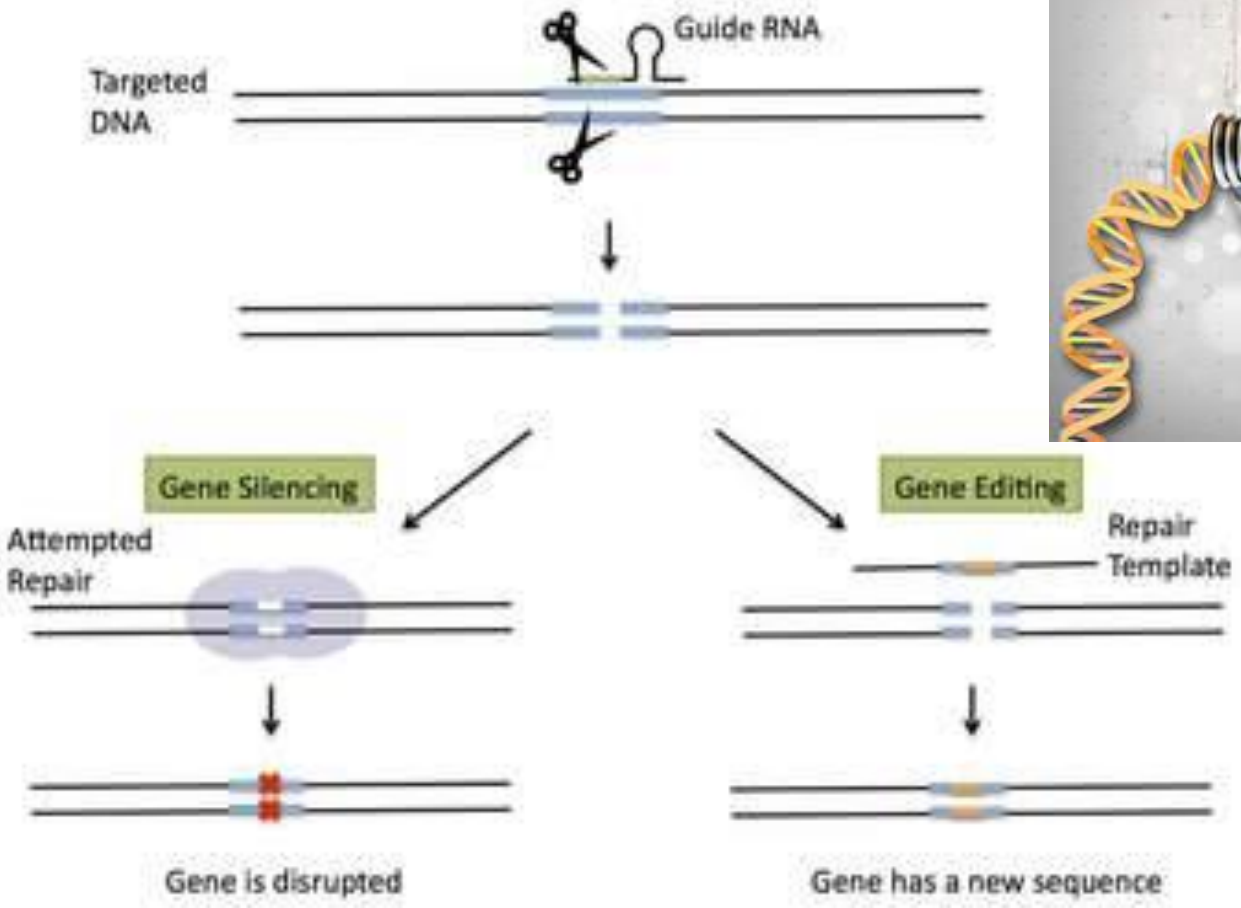
シャルパンティエ　　ダウドナ

ゲノム編集技術　CRISPR-Cas9


RuvC domain
HNH domain
C-term domain
Cas9
guide RNA
5′ –
3′ – UUUU
AAGGCUAGUCCGUUAUCAA
3′ –
5′ –
chromosome
PAM


Guide RNA
Cas9 enzyme
Matching gene sequence
enomic DNA


CRISPR
CRISPR
CRISPR-Cas9


Guide RNA
Targeted DNA
Gene Silencing
Attempted Repair
Gene is disrupted
Gene Editing
Repair Template
Gene has a new sequence

機械の知能の計算主義モデル

計算主義とは何か？

人工知能論における「計算主義」とは、人間の「知能」の本質を「計算」として抽象しようとするものである。僕は、基本的には、「計算主義」の立場に立っている。

「知能」に対する「計算主義的」なアプローチは、僕が人工知能が将来実現すべき課題として関心を持っている、人間の「言語能力」や日常的な「論理的推論能力」、抽象的な「数学的能力」、さらには「科学する能力」の機械による実現には、そうしたアプローチが不可欠だと考えている。

この時、「感情」や「意識」等々の機械による知能実現にまつわる、様々な問題を、当面は括弧に入れて保留しているのだが。

もちろん、人間の「知能」にまつわる問題を、一つの方法論で対応するのは難しい。それについては、別の機会に述べようと思う。

計算主義の含意

「計算」を、ある入力をある出力に変換する「機能」であると考えれば、同一性を残したまま、その具体的実装は捨象できる。ある計算が、そろばんの上でなされようと、スパコンの上でなされようと、はたまた、「計算」を担うのが、タンパク質だろうがシリコンだろうが、していることは「同じ」であると考えることはできるのである。

機械と人間の「知能」の本質を「計算」として抽象することで、その抽象の限りでは、「機械」と「人間」は等値されうる。だから、この抽象のもとでは、機械ができることは人間にもできることになり、人間ができることは機械にもできることになる。特に、後者の「人間ができることは機械にもできる」という主張は、人工知能技術の可能性を考える上では、とても強い主張になる。人工知能の可能性を考える上では、「我々自身を見よ！」という以上の強いメッセージを、僕は、思いつかない。

コンピュータが数学的定理を証明することの意味

最近、僕の友人がカードゲーム「大貧民」の必勝法についての定理を、コンピュータで証明して見せた。GitHubで公開された証明を、僕もダウンロードして実行できた。

> 僕のマシンは、ちゃんと証明できました！
> だから、僕も「証明」できたんでしょうね。
> マシンの助けを借りて証明の「正しさ」を「信じる」
> ことができると言うのは、面白いことです。

とFacebookに書いた。
こうした議論に、多少の違和感を持った人も多いと思う。
その違和感の正体を考えることは、いい思考実験になると思う。

コンピュータが数学的定理を証明することの意味

「コンピュータは証明なぞしていない。証明を作ったのは人間。」

確かに。

ここで利用された Coqは「証明支援システム」と呼ばれるもので、問題を与えると証明を返す「自動証明システム」ではない。今回の例の場合、Coqのプログラムの形で証明を書いたのは僕の友人である。

「コンピュータは与えられた証明を機械的に実行しただけ。」

それはそうだ。

ただ、そう考えてもらえると、僕は嬉しい。

「証明」=「プログラム」=「計算」

数学的証明がコンピュータで機械的に実行可能なプログラムの形で記述できるという認識は、比較的新しいものだ。

理論的には、こうした認識は、1970年代のハワードの「カリー・ハワード対応」の(再)発見と、それに基づくマーティン・レフの「従属型理論」にさかのぼるものだ。

いまから 40年近く前に、論理学者と計算機学者の一部は、「証明」=「プログラム」という認識にたどり着いていた。ただ、現代人の多くは、「プログラムは証明である」とは考えていないように思う。

「証明」=「プログラム」=「計算」

実は、「計算可能性」や「証明可能性」の探究を通じて、「計算」と「証明」は同じ概念であるということに、今から90年近く前のゲーデルやチャーチやチューリングは気がついていた。

「計算」と「証明」は、それらにチューリングが与えたモデルである「チューリング・マシン」を見ればわかるように、その振る舞いは、機械的な過程として定義できる。だから、「証明」が機械的に実行可能なのは当然である。

ただ、その当時には、「コンピュータ」も「プログラム」も、影も形もなかったのだ。

機械は、論理的＝数学的推論能力を持つ

コンピュータが普及し「人工知能」についての議論が活発な現在、確認すべき重要なテーゼがあると僕は考えている。それは、論理的＝数学的能力が機械的に定義されうるのだから、機械は、それ自身、論理的＝数学的推論能力を持つということである。

この主張は、論理的＝数学的推論が機械的なモデルを持つということを、機械を主語として言い換えただけである。

「計算機は計算能力を持つ」ということを疑う人は（現在では）いない。

そうした確信を支えているのは、多くの人が計算するのに、日常的に計算機を使っているからである。しかし、たとえ、電卓やExcelであっても、それらが存在しなかった時代の人には、それらが示す計算能力は、信じられない魔法のように見えるに違いないと僕は思う。

機械は、論理的＝数学的推論能力を持つ

数学的証明も論理的＝数学的推論も、機械的に実行可能である。

そうした確信は、今のところ、Turingマシンの構成や「型の理論」に基づいた「理論的」なもので、大多数の人の実感からは、はなれているかもしれない。

ただ、それは、まだ多くの人が、数学的証明や論理的＝数学的推論に、機械を使っていないことの反映だと思う。いづれ、人々の意見は大きく変化して、数学的証明も論理的＝数学的推論も、機械的に実行可能であると考えるようになると僕は信じている。

知能の「ディープ・ラーニング」モデルとの違い

一つ注意しておきたいのは、こうした機械の持つ論理的＝数学的推論能力への注目は、現在の人工知能技術の中心的な手法である、「ディープ・ラーニング」とは異なる「知能」へのアプローチだということである。

「ディープ・ラーニング」は、基本的には、大量のデータから何かを推論する。それは、帰納的推論の一種である。そこでは「データ」から「学ぶ」ことが、本質的に重要である。だから「ラーニング」なのだ。確かに、それは人間の「知能」や「知識獲得」の一面を捉えている。

知能の「ディープ・ラーニング」モデルとの違い

ただ、論理的＝数学的推論には、単純化して言えば、データは必要ないのだ。1＋1＝2 を証明するのに、経験的データは不要である。

実践的に重要なことは、「ディープ・ラーニング」では、数学の証明はできないということである。それは、我々の「知能」の重要な一部である、論理的＝数学的推論にモデルを提供することができない。

複雑性理論の知見

複雑性理論とは何か？
人間と機械の知能の限界を考える

20世紀、テクノロジーの分野では、コンピュータの登場によって、人間と機械の計算能力は飛躍的に発展した。それは、多くの人の日常生活を大きく変えた革命的な出来事であった。

20世紀の前半、コンピュータがまだ存在しなかった時期に、すでに、ゲーデル、チューリング、チャーチらによって、「計算とは何か」「どのような計算が実際には可能であるか」という問題の探究が始まっていた。

人間と機械の知能の限界自身を数学的分析の対象とする

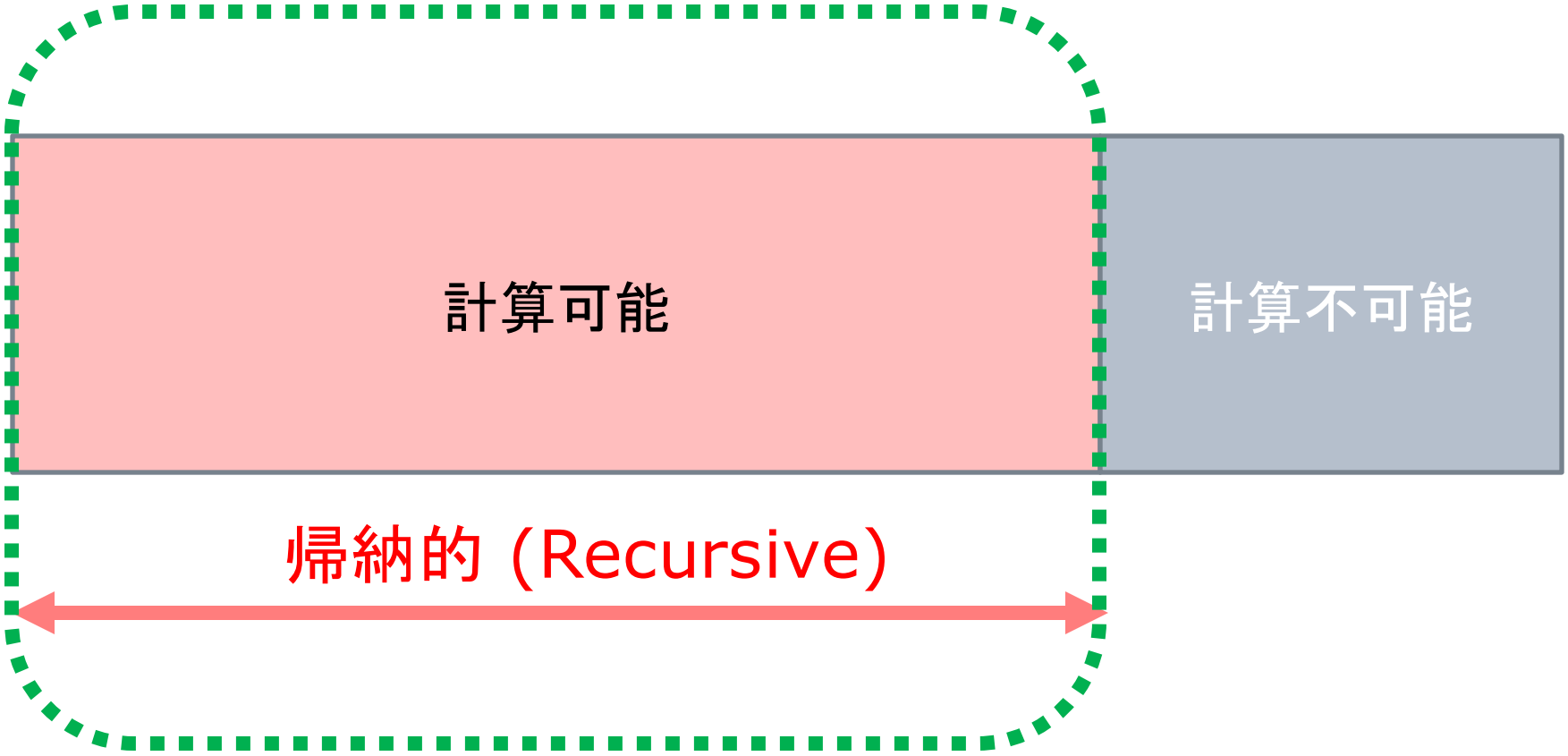
それは、「計算＝証明できないものが存在する」という発見を中心に、20世紀の科学・哲学の分野に革命的なインパクトを与えた。それが、今日では計算科学の一分野として発展している、「計算複雑性理論」の起源である。

人間の知能の中核に計算能力があるのなら、計算複雑性理論は、人間と機械の知能の限界自身を数学的分析の対象としたものであると考えることができる。

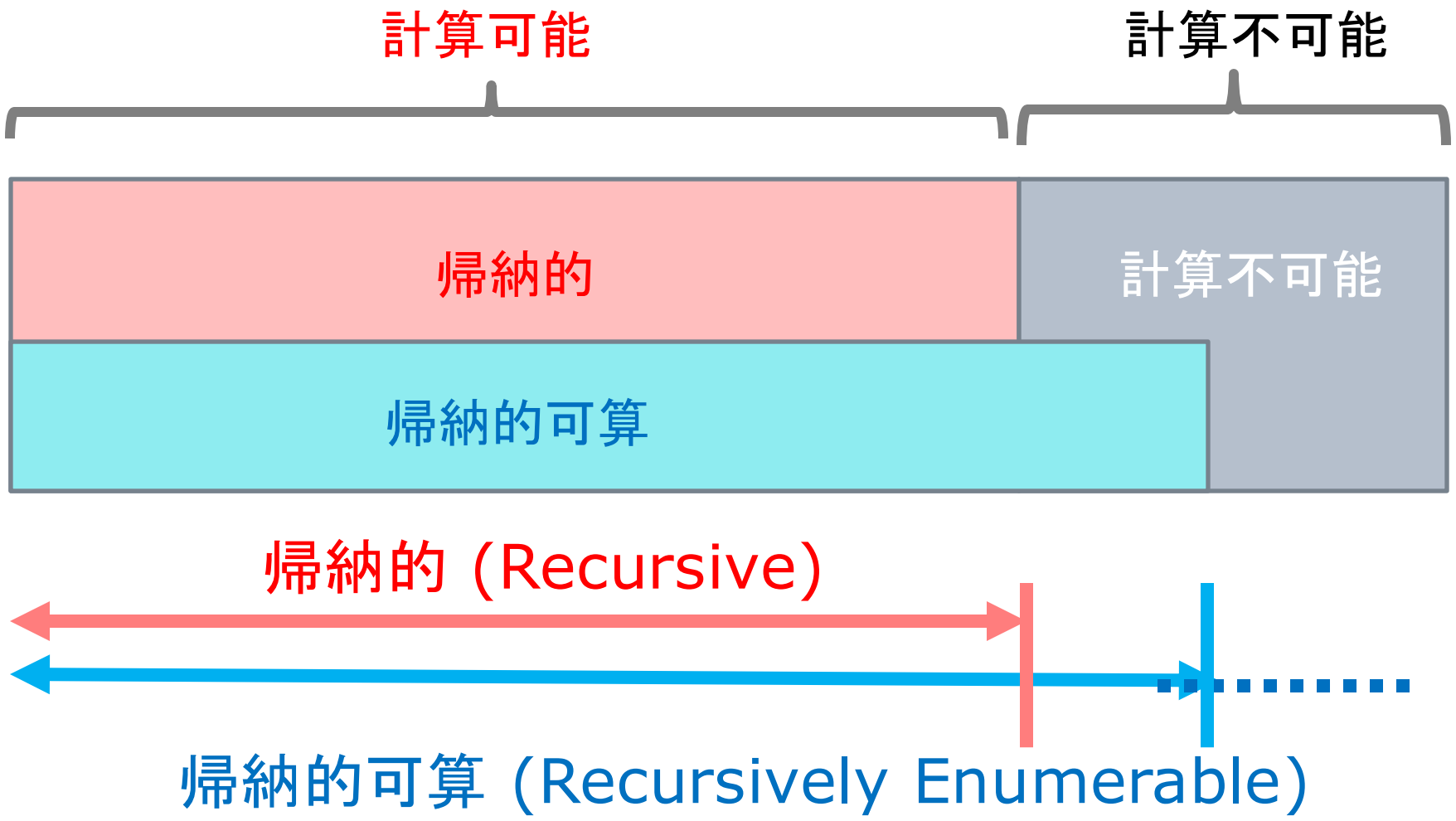
20世紀の複雑性理論の達成

Church-Turing Thesis

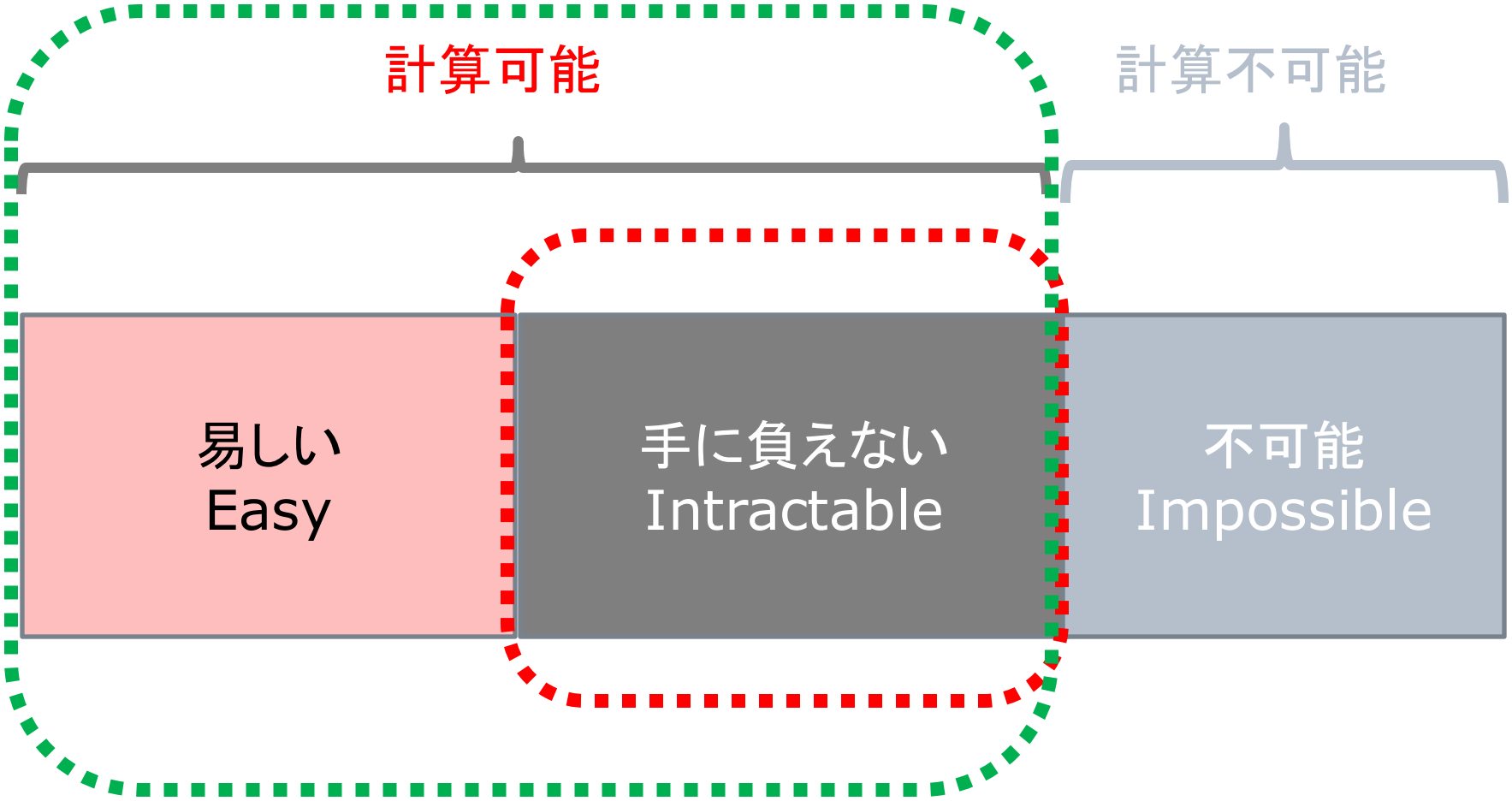
計算可能なものは帰納的である

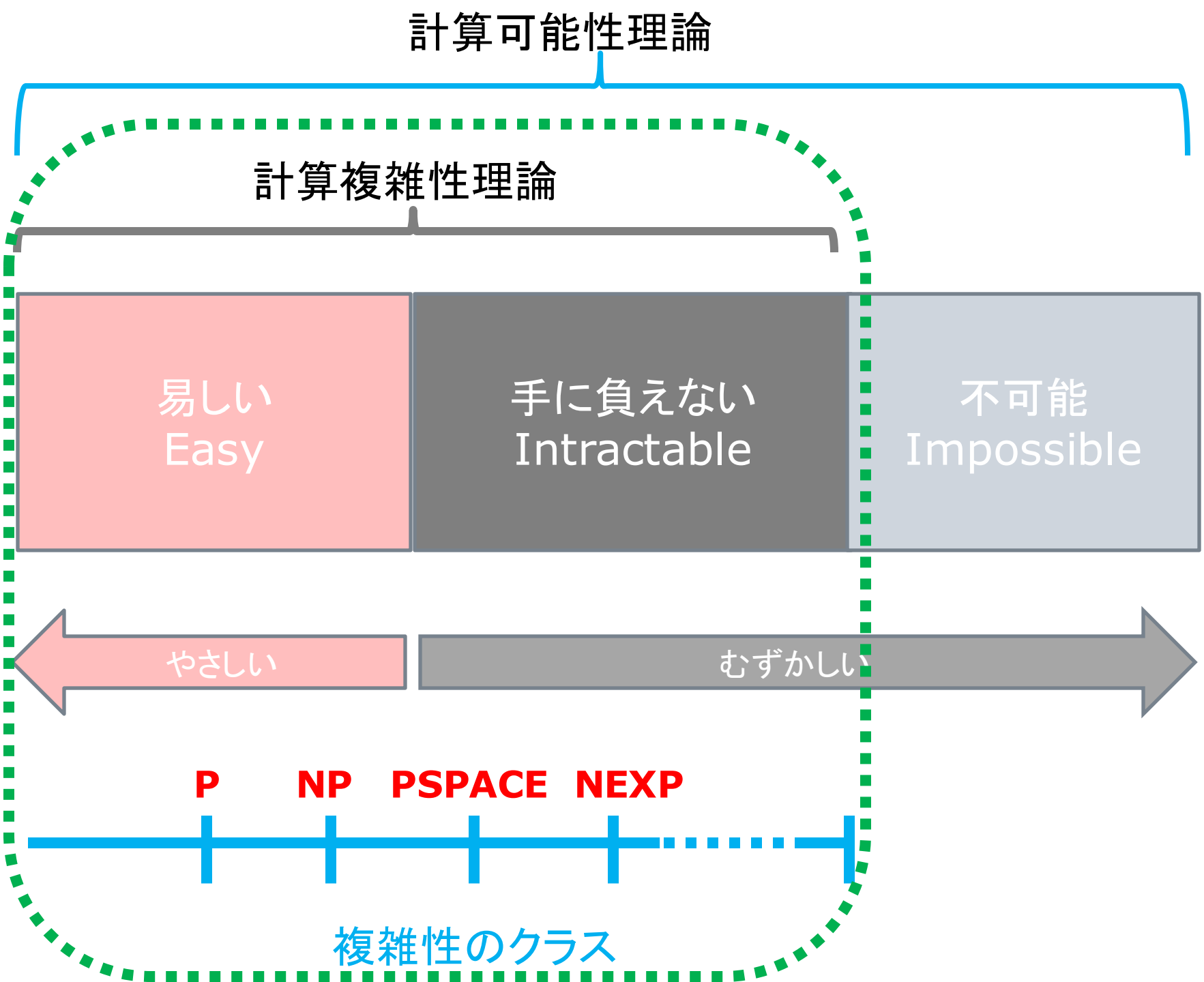


帰納的と帰納的可算


計算可能
計算不可能
帰納的
計算不可能
帰納的可算
帰納的 (Recursive)
帰納的可算 (Recursively Enumerable)

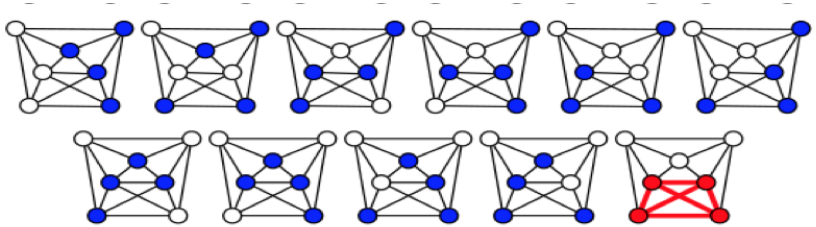
計算可能だが計算が手に負えないものがある



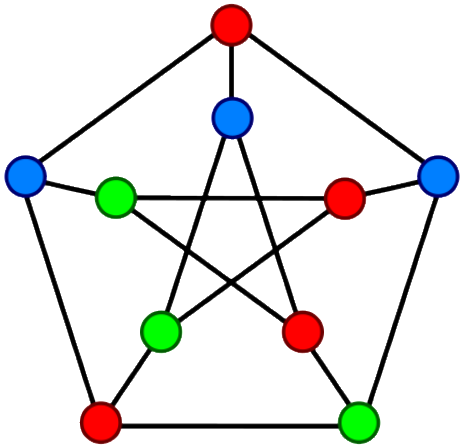

計算可能性理論
計算複雑性理論
易しい
Easy
手に負えない
Intractable
不可能
Impossible
やさしい
むずかしい
P
NP
PSPACE
NEXP
複雑性のクラス

NP-完全問題

- 1971年、Cookは、充足可能性問題が、NP-完全であることを証明した。ここに初めて、NP-完全というコンセプトが登場する。（同じ頃、ソヴィエトの数学者Levinも同じ結果を得ている）
  現代の複雑性理論は、この発見から始まる。
- 1972年、Karpは、ある問題がNP-完全であることを証明する標準的な「還元」の手法を開発し、多くの問題がNP-完全であることを示した（「Karpの21の問題」）。
- このNP-完全問題のリストは、現在では数百の問題をカバーして拡大している。（例えば、”List of NP-complete problems” https://goo.gl/XUSvRz ）
- これらの問題の「手に負えなさ: “Intractability ”」は、例外的なものではなく、一般的なものである。
- Cookは1982年、Karpは1985年、Turing賞を受賞した。



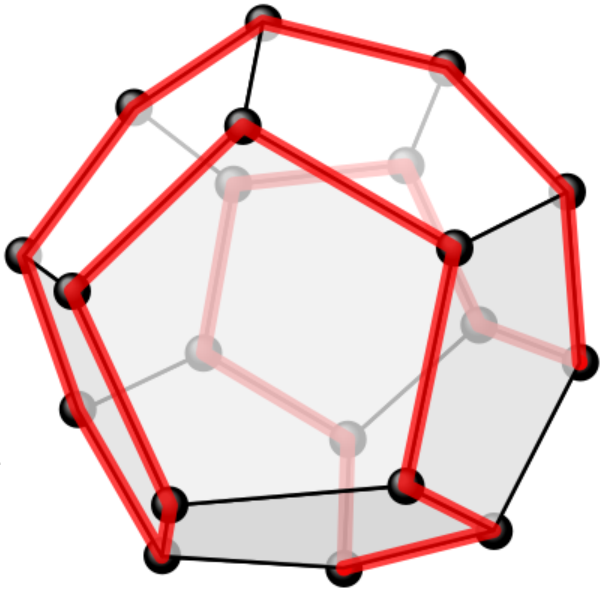
Clique



Graph Coloring

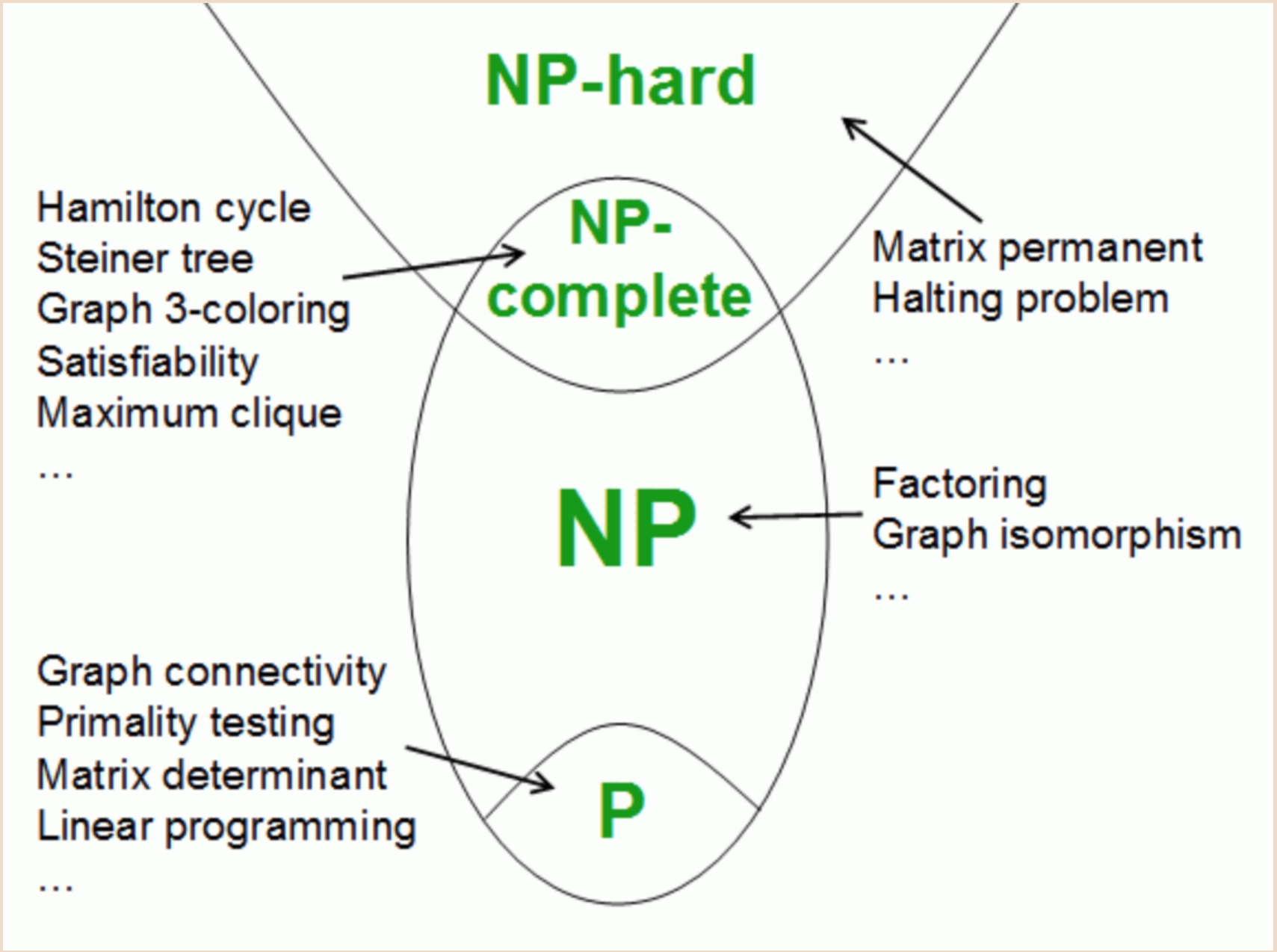
$$(l_1 \vee l_2 \vee x_2) \wedge$$
$$(\neg x_2 \vee l_3 \vee x_3) \wedge$$
$$(\neg x_3 \vee l_4 \vee x_4) \wedge \cdots \wedge$$
$$(\neg x_{n-3} \vee l_{n-2} \vee x_{n-2}) \wedge$$
$$(\neg x_{n-2} \vee l_{n-1} \vee l_n)$$

3-SAT



Hamilton Path

様々なNP-完全問題


NP-hard
NP-complete
NP
P
Hamilton cycle
Steiner tree
Graph 3-coloring
Satisfiability
Maximum clique
…
Matrix permanent
Halting problem
…
Factoring
Graph isomorphism
…
Graph connectivity
Primality testing
Matrix determinant
Linear programming
…

21世紀の複雑性理論の二つの達成

- Googleによる量子超越性の実証
- Natarajanらの MIP*=RE 定理

参考資料
「マルゼミ「Interactive Proof 入門」に向けて」
https://www.marulabo.net/docs/math-foundations/

計算複雑性をめぐる最近の二つの発見
1. Googleによる量子超越性の実証

- 21世紀に入って、複雑性理論は、新しい進展を見せる。特に、去年今年と相次いで、重要な発見が相次いだ。この二つの発見は、人間と機械との知能の限界をめぐって、科学と哲学、特に人工知能論に大きな影響を与えるのは避けられないと、僕は考えている。
- 第一のものは、2019年のGoogleによる「量子超越性」の実証である。複雑性の理論の領域では、既に、ショアのアルゴリズムの発見前後に、計算能力において量子コンピュータは古典的なコンピュータのをうわまわるという理論的主張は確立していた。
- Googleの実験は、そうしたマシンを実際に作って見せたことで、その理論が実際に成り立つことを示したという意味で画期的なものである。

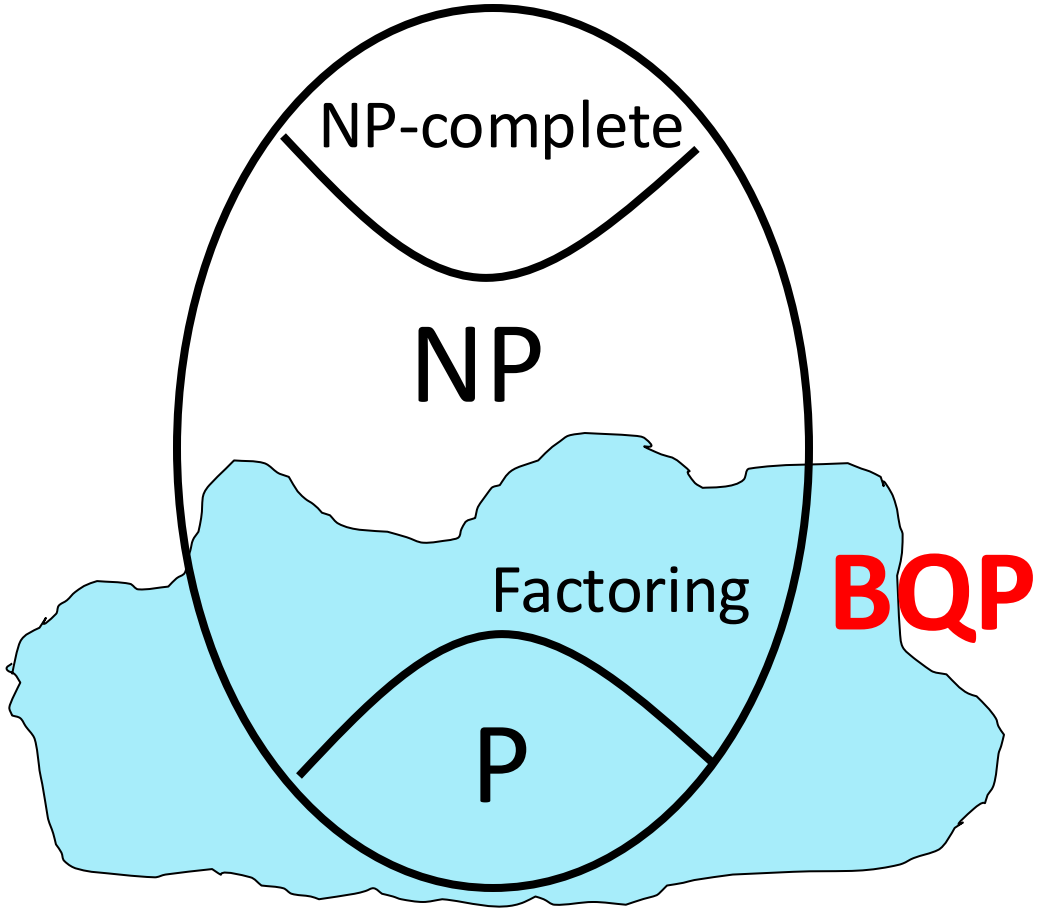
計算複雑性をめぐる最近の二つの発見
2. NatarajanらのMIP*=RE 定理

- 第二のものは、2020年のNatarajanらによる MIP*=RE 定理の証明である。
- 彼らは、エンタングルメントした量子を共有する量子コンピュータが、強力な計算パワーを持つことを示すとともに、機械と人間との間のインタラクティブな対話が、その計算パワーを引き出しうることを示した。

Googleによる量子超越性の実証

BQP (Bounded-Error Quantum Polynomial-Time): The class of problems solvable efficiently by a quantum computer, defined by Bernstein and Vazirani in 1993

Shor 1994: Factoring integers is in **BQP**



量子超越性 は、「拡張されたChurch-Turing Thesis」が成り立たないことを示す

- Scott Aaronsonは、Googleによる量子超越性の達成を、次のような定式化での「拡張されたChurch-Turing Thesis」が成り立たないことを実験的に示したものだと評価した。
- なぜなら、Googleの実験は、量子コンピュータが古典的コンピュータ（その計算能力はTuringマシンに等しい）を超える計算能力を持つことを示したからだ。

拡張されたChurch-Turing Thesis

物理的な世界で計算されるものは、
Turingマシンによって多項式時間で
計算されるものである

量子超越性 は、「拡張されたChurch-Turing Thesis」が成り立たないことを示す

- Scott Aaronsonは、Googleによる量子超越性の達成を、次のような定式化での「拡張されたChurch-Turing Thesis」が成り立たないことを実験的に示したものだと評価した。
- なぜなら、Googleの実験は、量子コンピュータが古典的コンピュータ（その計算能力はTuringマシンに等しい）を超える計算能力を持つことを示したからだ。

「拡張されたChurch-Turing Thesis」が成り立たないことの人工知能論への影響

「拡張されたChurch-Turing Thesis」が成り立たないという発見は、人工知能論に大きな意味を持ちうる。

「計算主義」的人工知能論は、原理的には、人間の知能が機械の計算能力に還元可能だと考える。その限りでは、機械と人間の能力は同等である。

量子コンピュータが、計算のモデルとしてのTuringマシンを上回る計算能力を持つのなら、機械としての量子マシンは、人間の知能を超える能力を持つことになる。

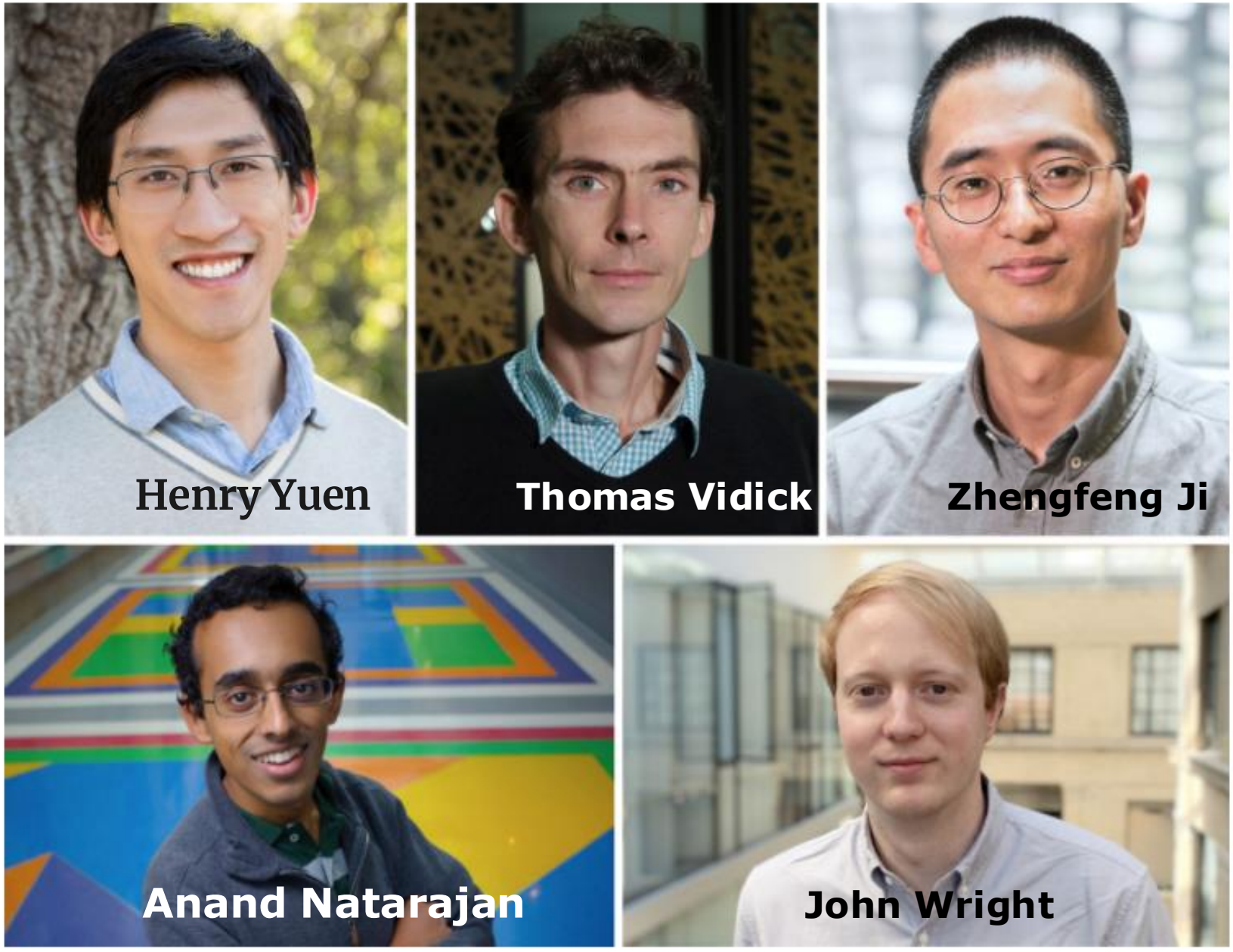
「拡張されたChurch-Turing Thesis」が成り立たないことの人工知能論への影響

誤解してはいけないのは、量子コンピュータが NP-完全問題や停止問題を解く能力を持つ訳ではないということである。

それでも、複雑性理論的には、BQPは「Turingマシンによって多項式時間で計算されるもの」より、強力な計算能力なのである。それは、人間には真似できないものだ。

MIP* = RE

“MIP*=RE“
Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, Henry Yuen
2020年1月
https://arxiv.org/pdf/2001.04383.pdf


Henry Yuen
Thomas Vidick
Zhengfeng Ji
Anand Natarajan
John Wright

論文 “MIP*=RE”

- **MIP*=RE :**
  We show that the class MIP* of languages that can be decided by a classical verifier interacting with multiple all-powerful quantum provers sharing entanglement is equal to the class RE of recursively enumerable languages.
- **Halting Problem :**
  An immediate byproduct of our result is that there is an efficient reduction from the Halting Problem to the problem of deciding whether a two-player nonlocal game has entangled value 1 or at most 1/2 .

論文 “MIP*=RE”

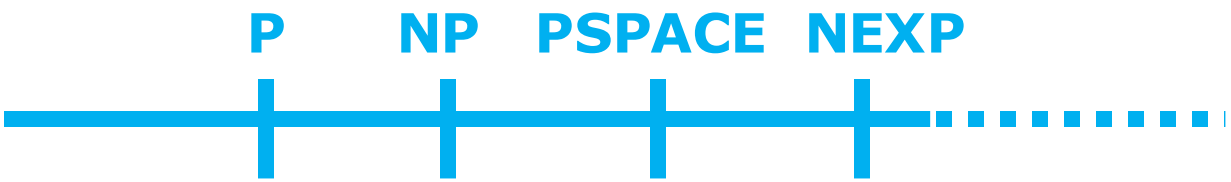
- **Tsirelson’s problem :**
  Using a known connection, undecidability of the entangled value implies a negative answer to Tsirelson’s problem: we show, by providing an explicit example, that the closure C_{qa} of the set of quantum tensor product correlations is strictly included in the set C_{qc} of quantum commuting correlations
- **Connes’ embedding conjecture :**
  Following work of (Fritz, Rev. Math. Phys. 2012) and (Junge et al., J. Math. Phys. 2011) our results provide a refutation of Connes’ embedding conjecture from the theory of von Neumann algebras..

MIP*=RE 以前

ここに断絶があった

計算複雑性理論

計算可能性理論

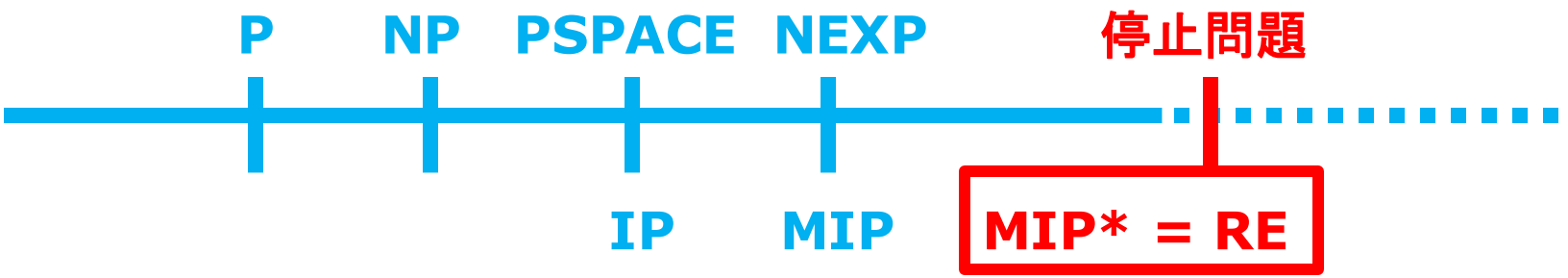


複雑性のクラス

MIP*=RE 以降の

計算複雑性理論

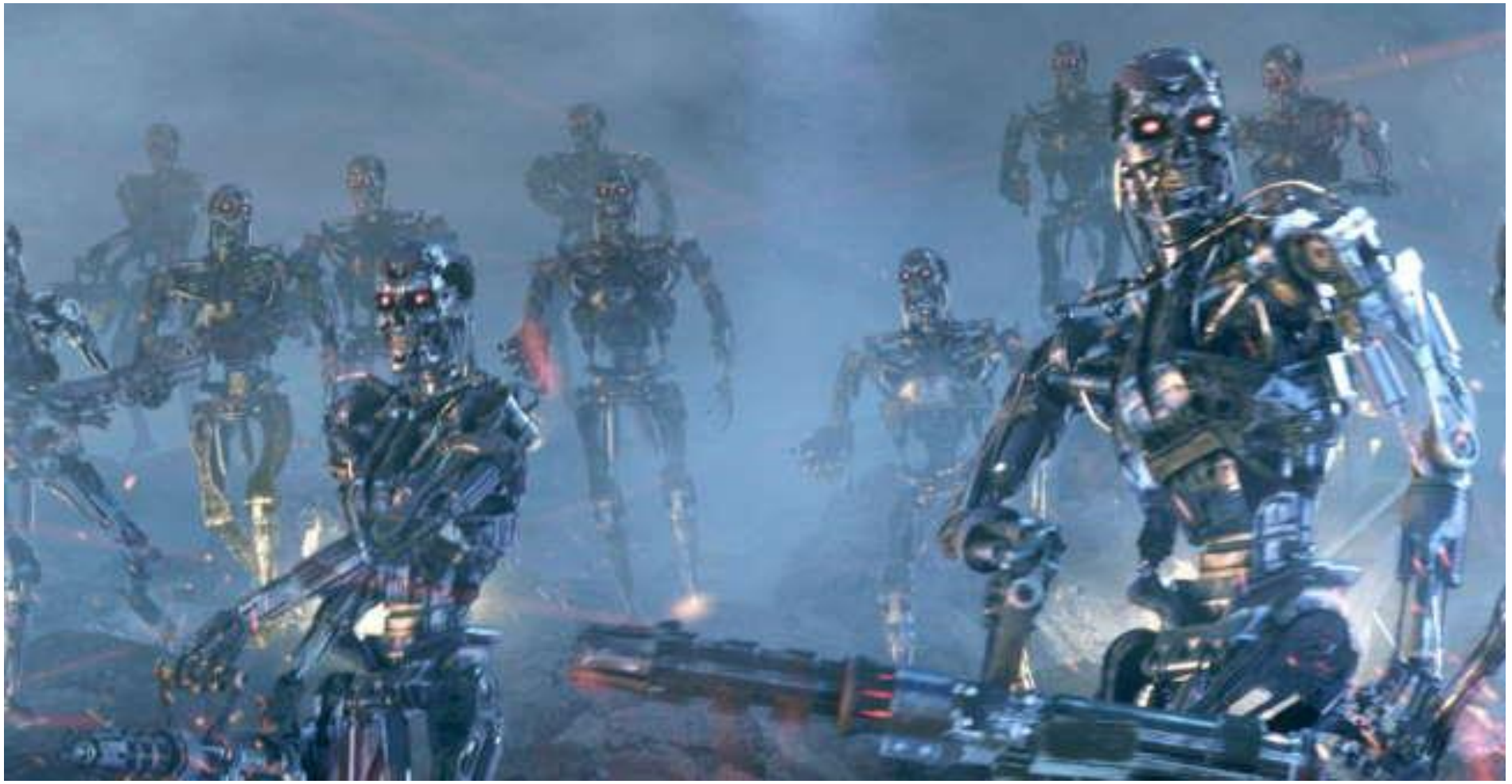
計算可能性理論



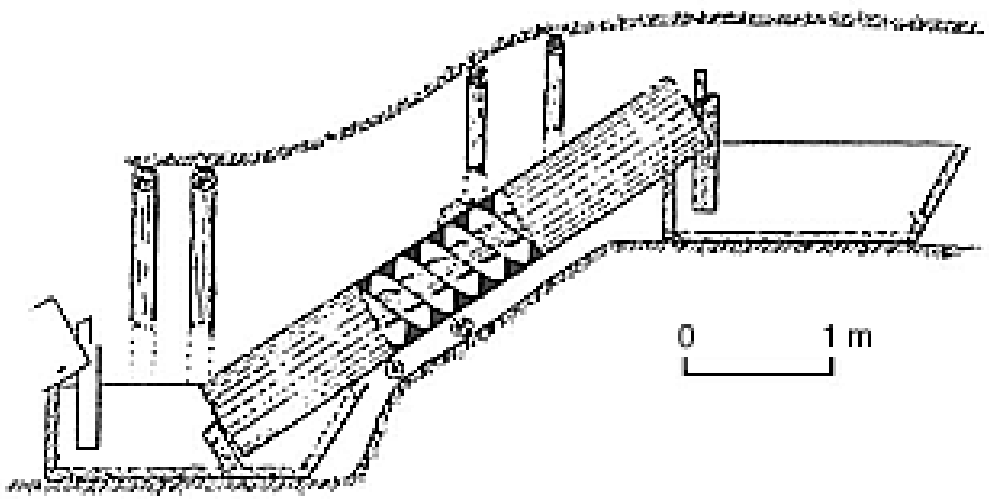
計算科学　　量子力学　　純粋数学

MIP*＝RE定理の射程

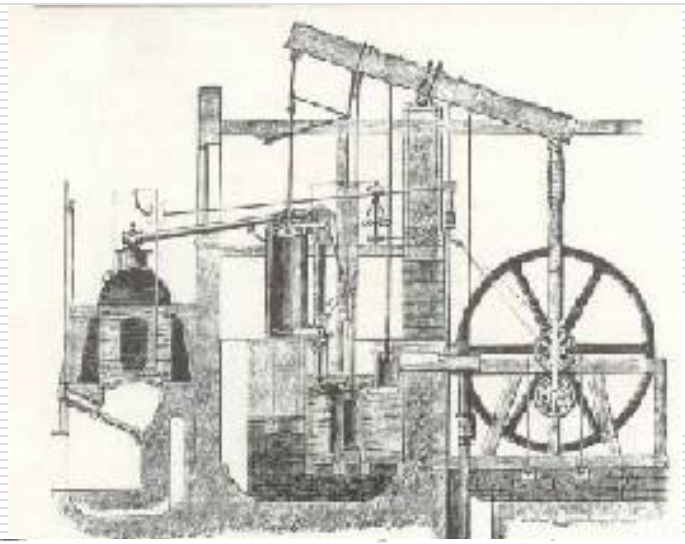
人間と機械の「共生」について



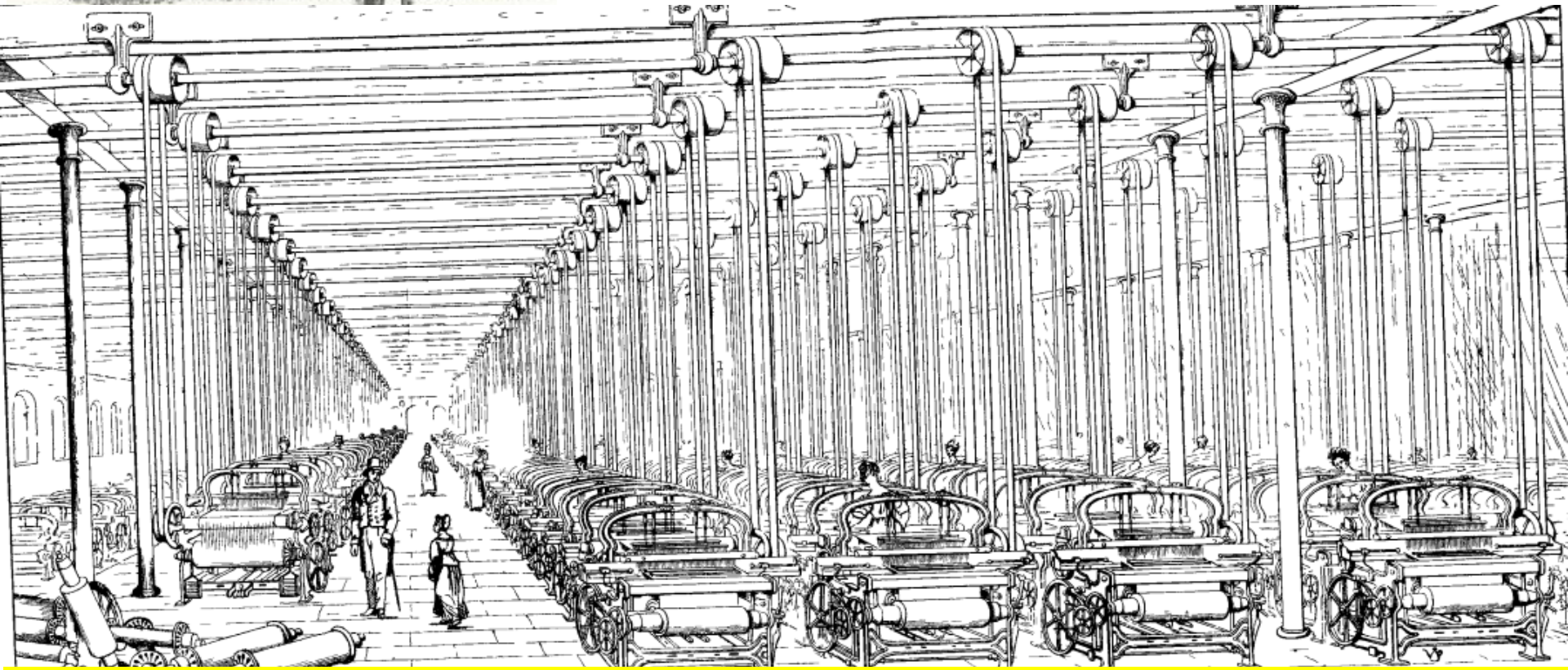
機械の進化



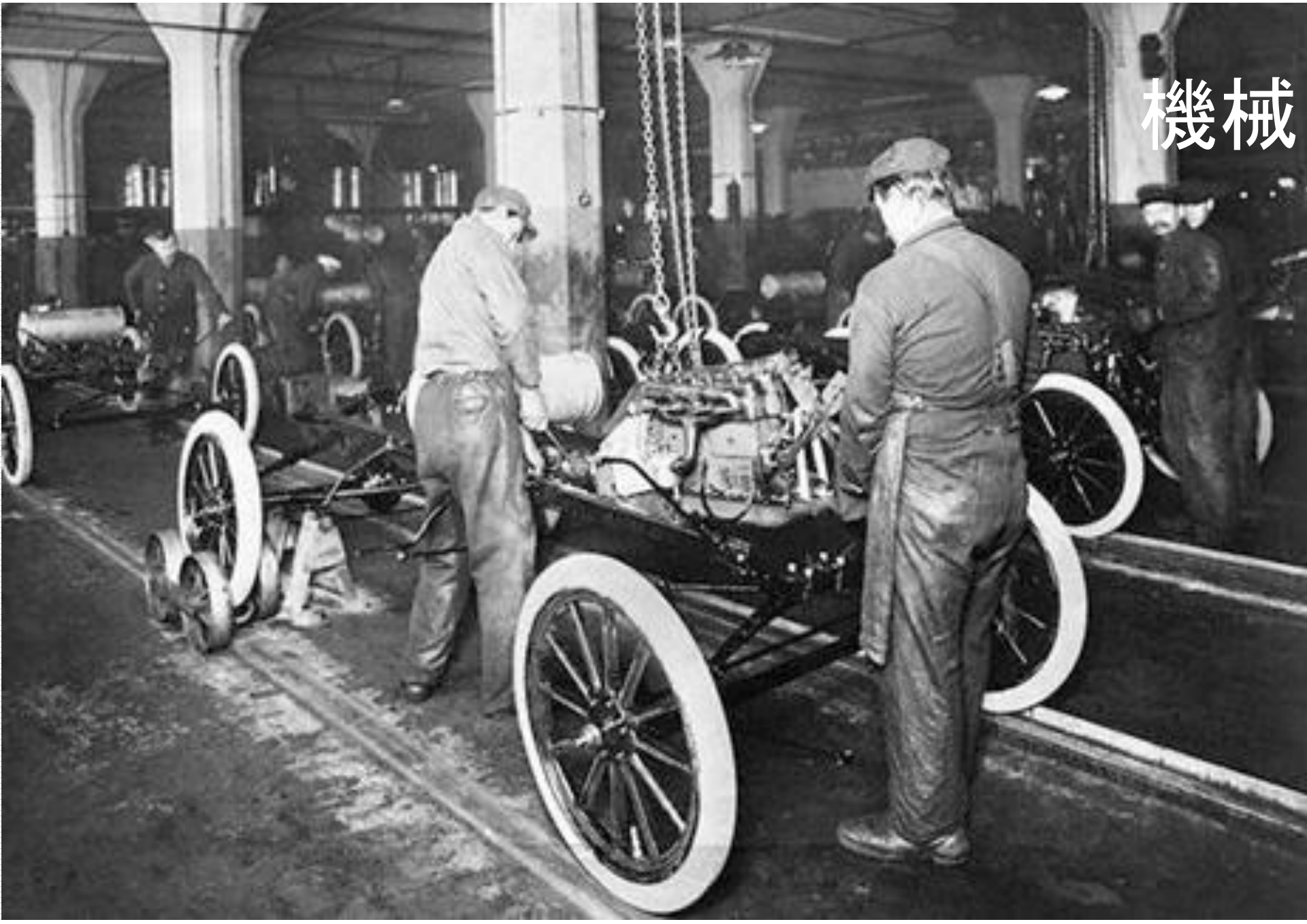
機械

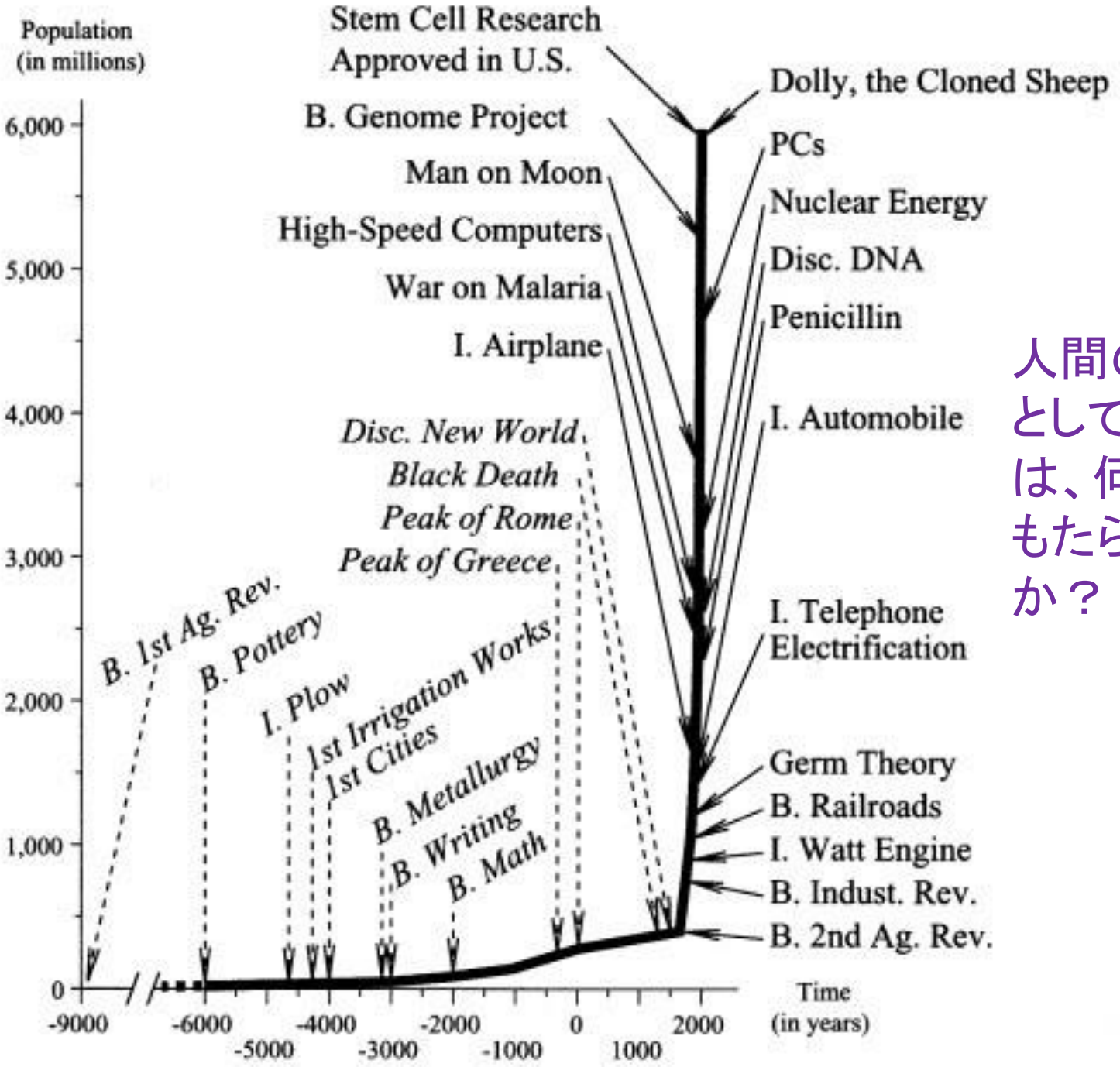


機械



http://socserv.mcmaster.ca/econ/ugcm/3ll3/ure/PhilosophyManufactures.pdf


機械



人間の生物種としての「成功」は、何によってもたらされたのか？

人間の感覚能力の拡大

感覚は、生物が外界を認識するために、進化を通じて発達させてきた能力である。

人間は、生物学的な進化以外の方法で、感覚の拡張を果たしてきた。こうした拡張は、主要には、20世紀においてなされた。

重要なことは、こうした機械による人間の感覚能力の拡大を導いたのは、科学的・数学的な認識能力だったということである。

自然は、数学という言葉で
書かれた書物である。

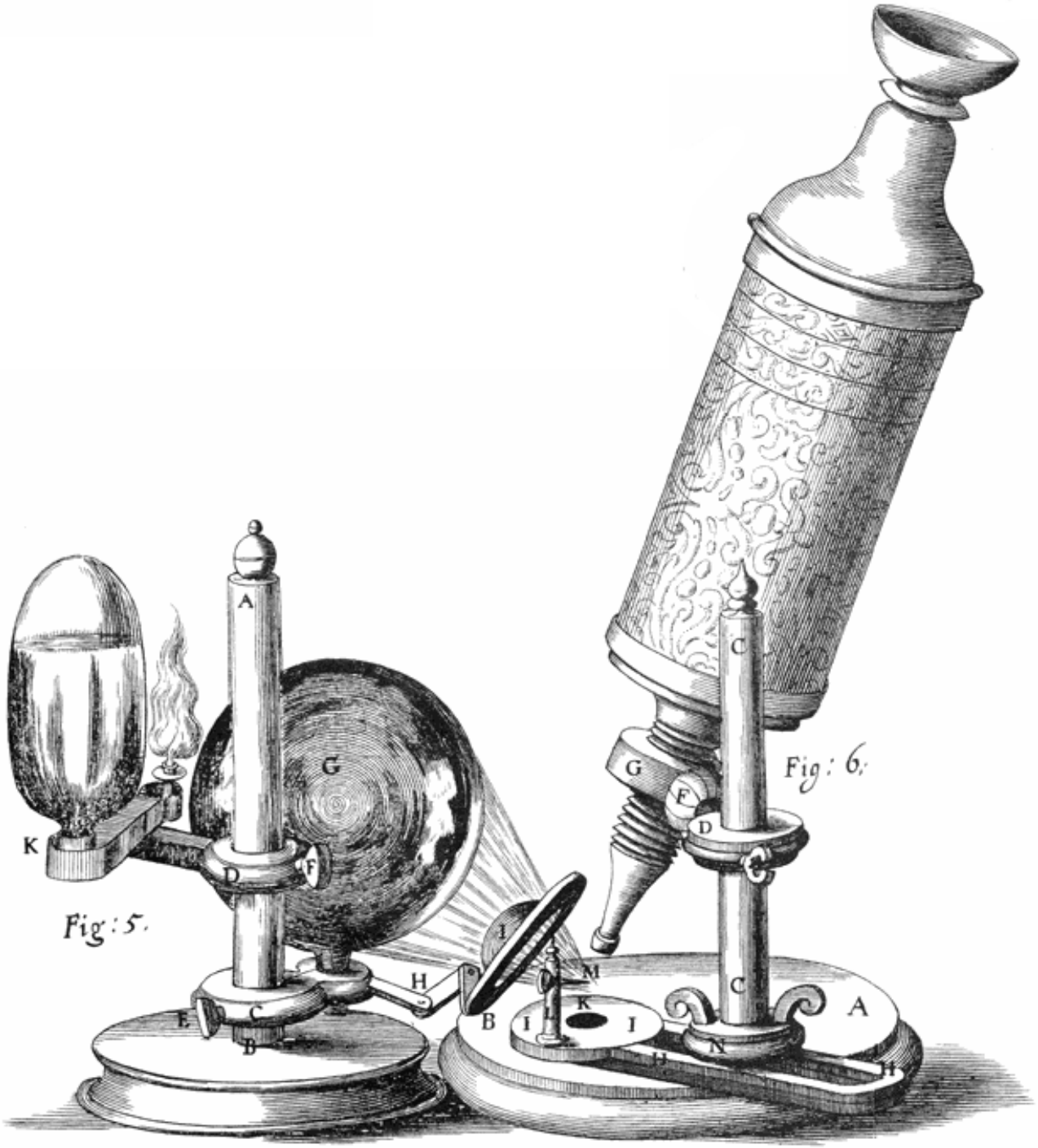
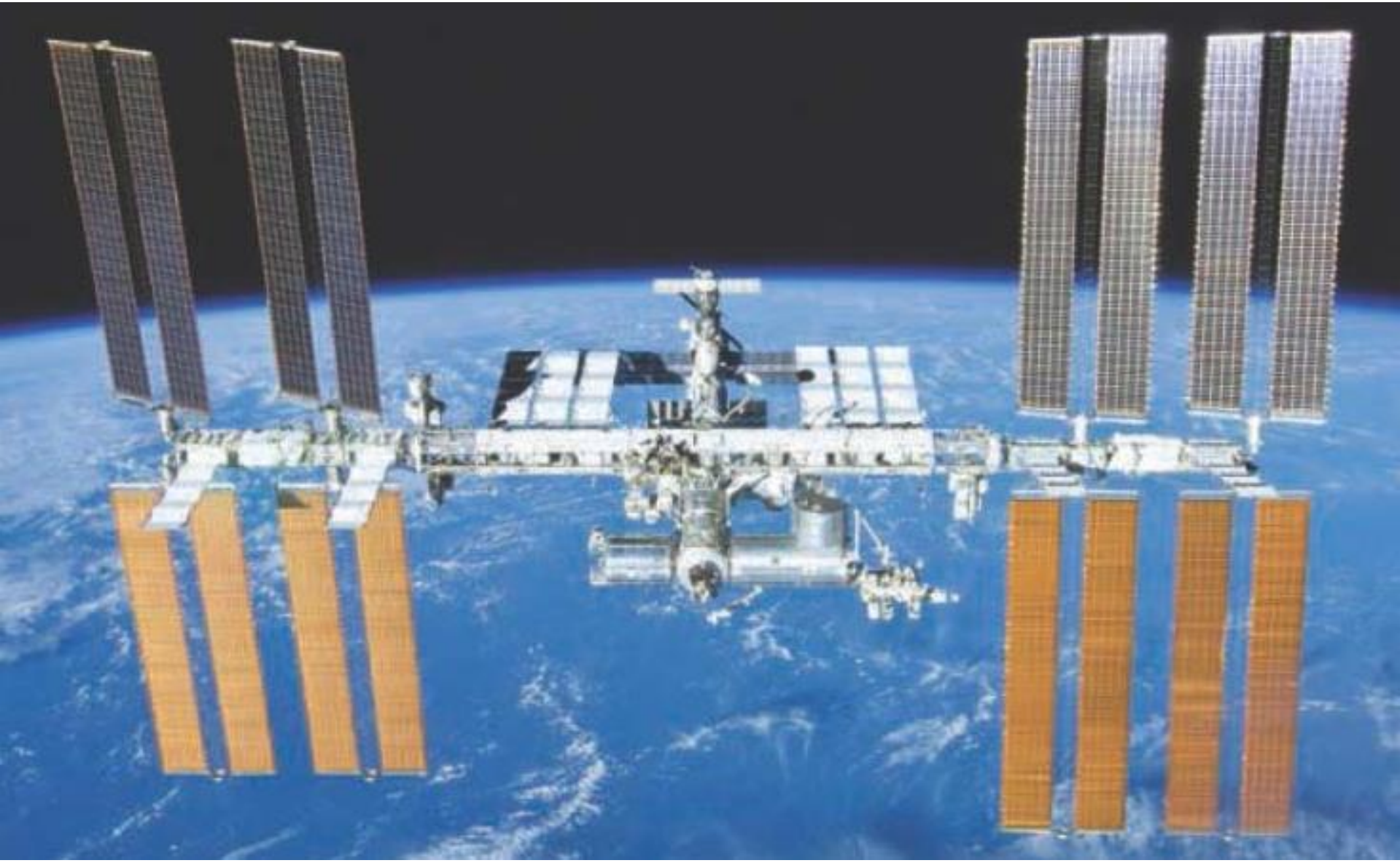
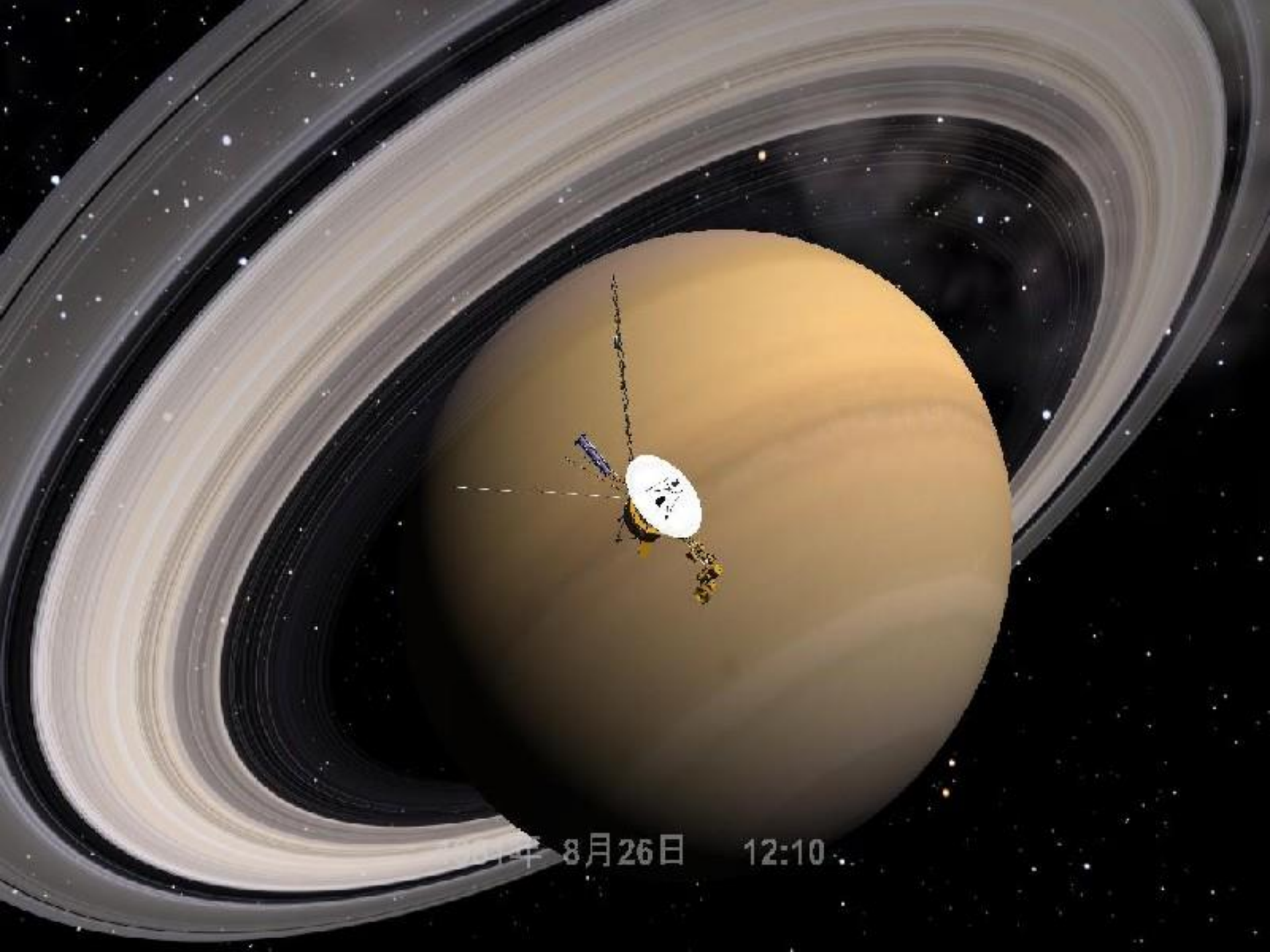


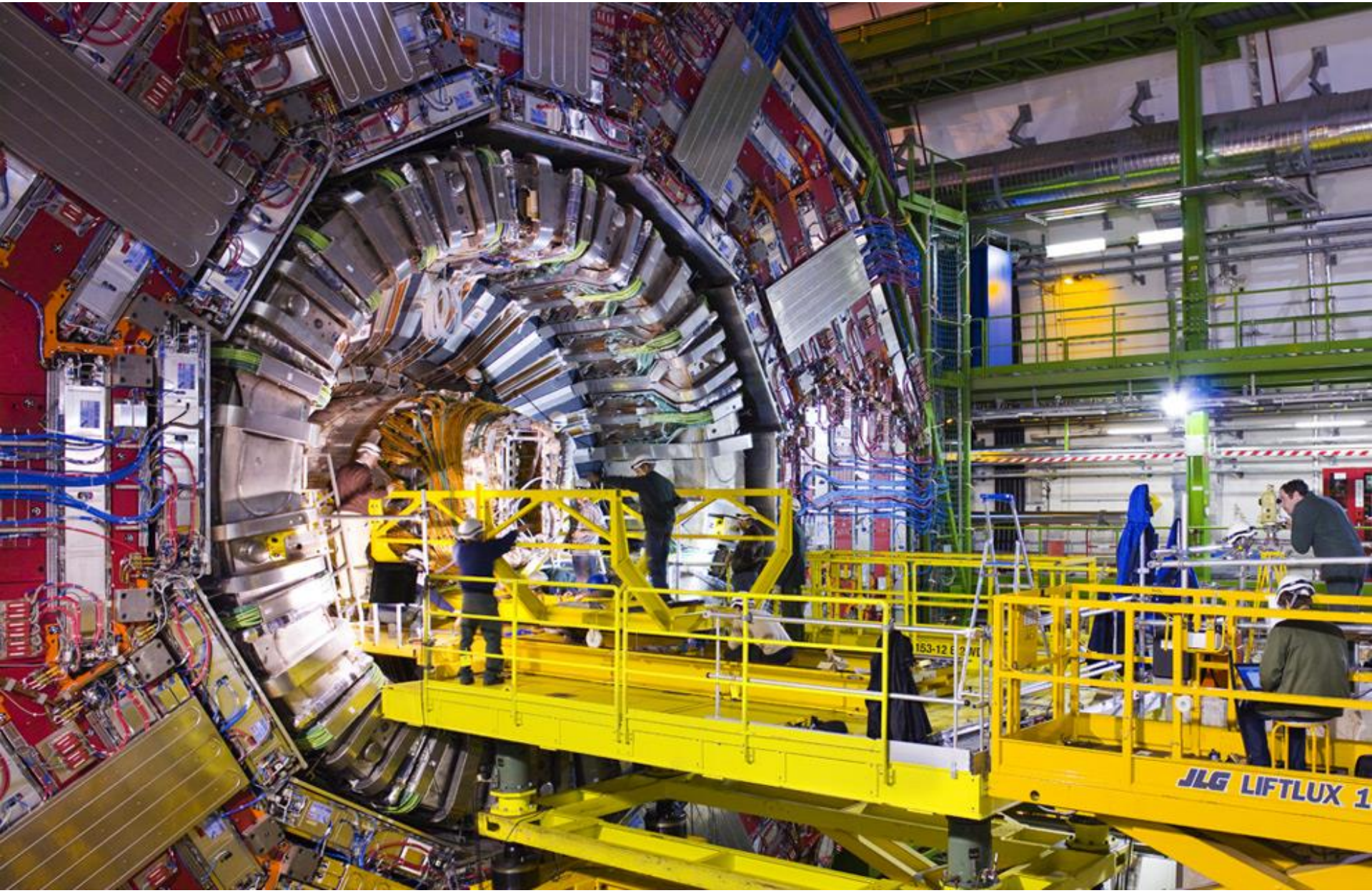
Fig: 5.

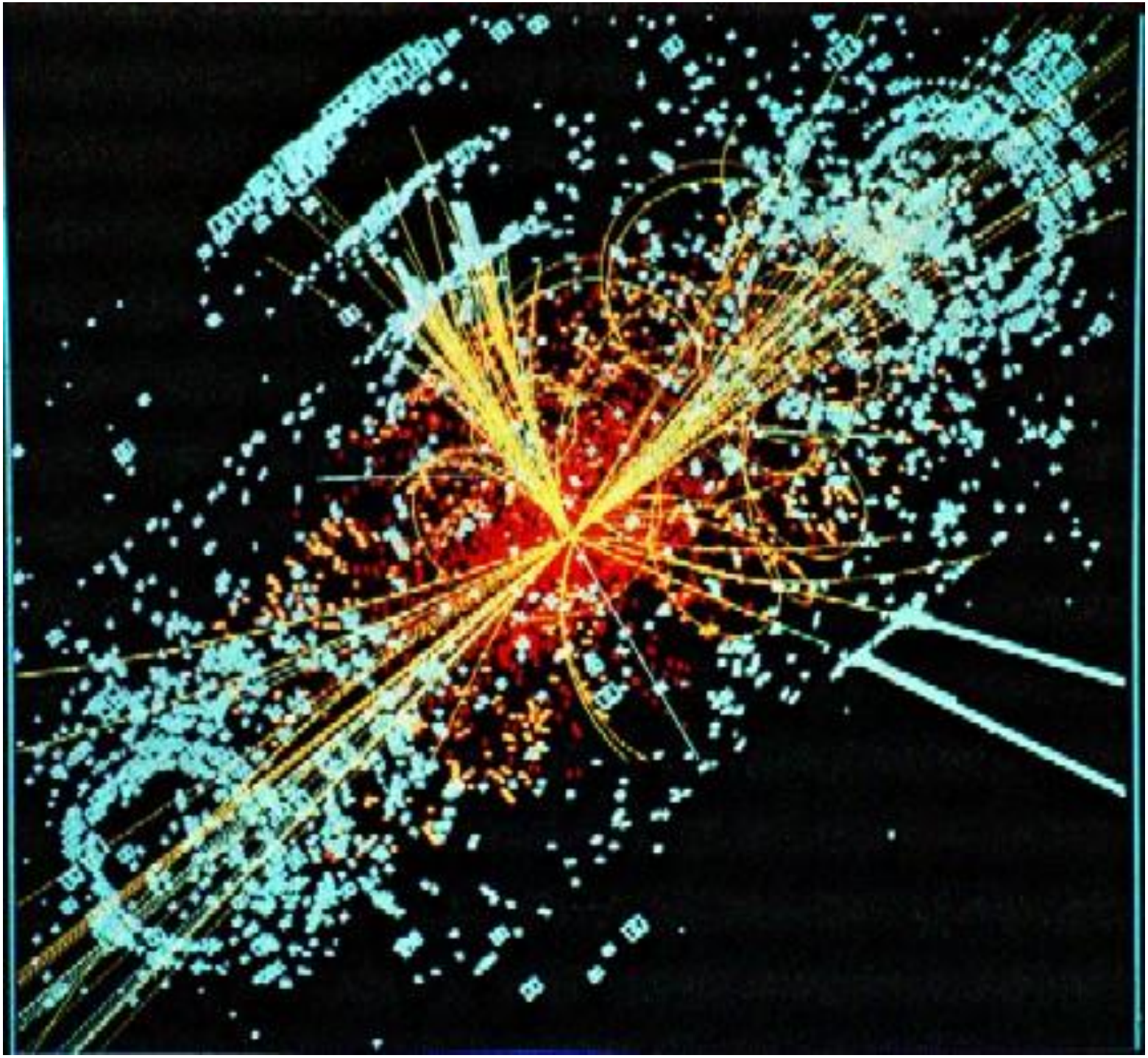
Fig: 6.

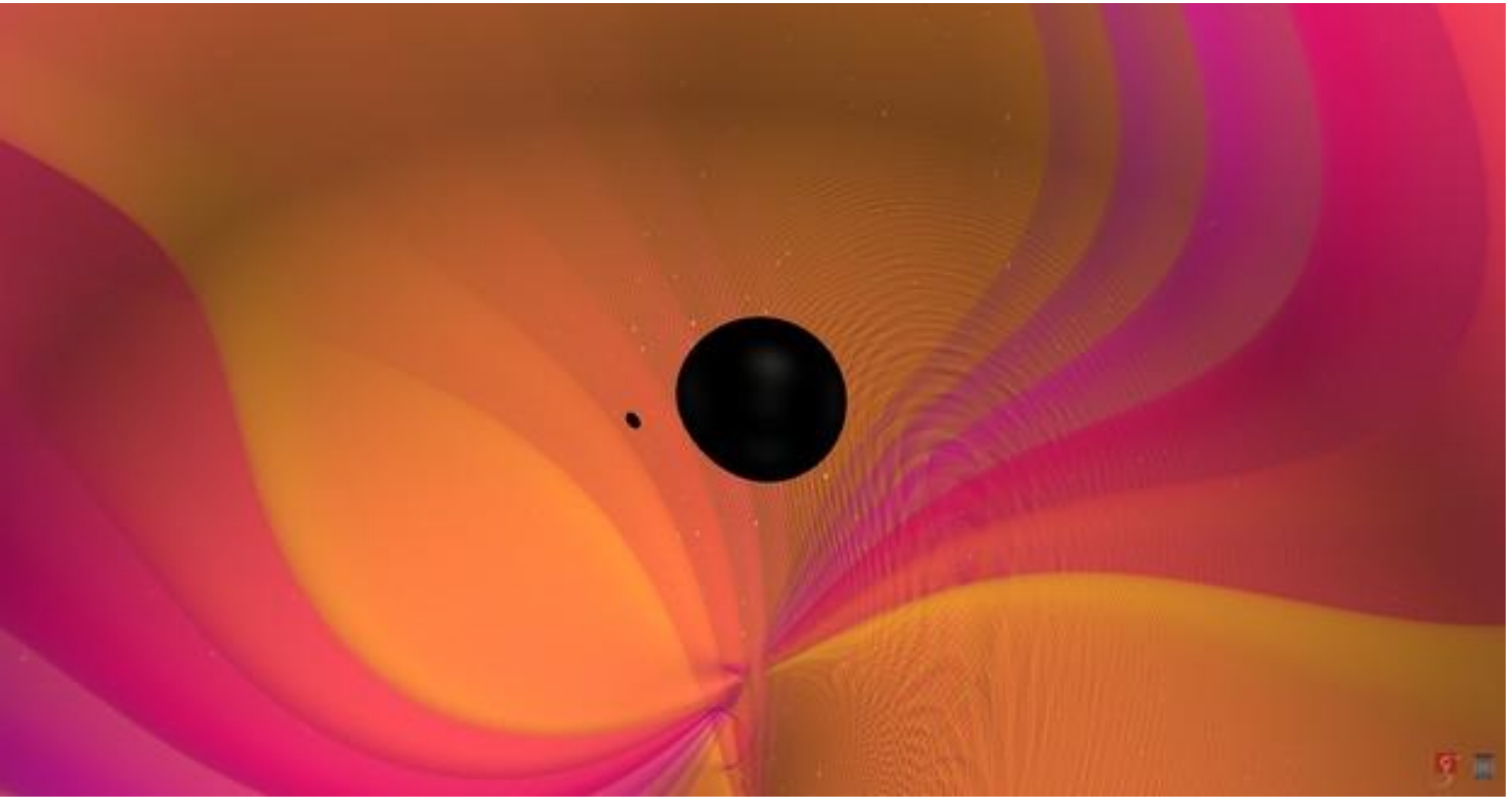

0309 102640 STS109 730 040

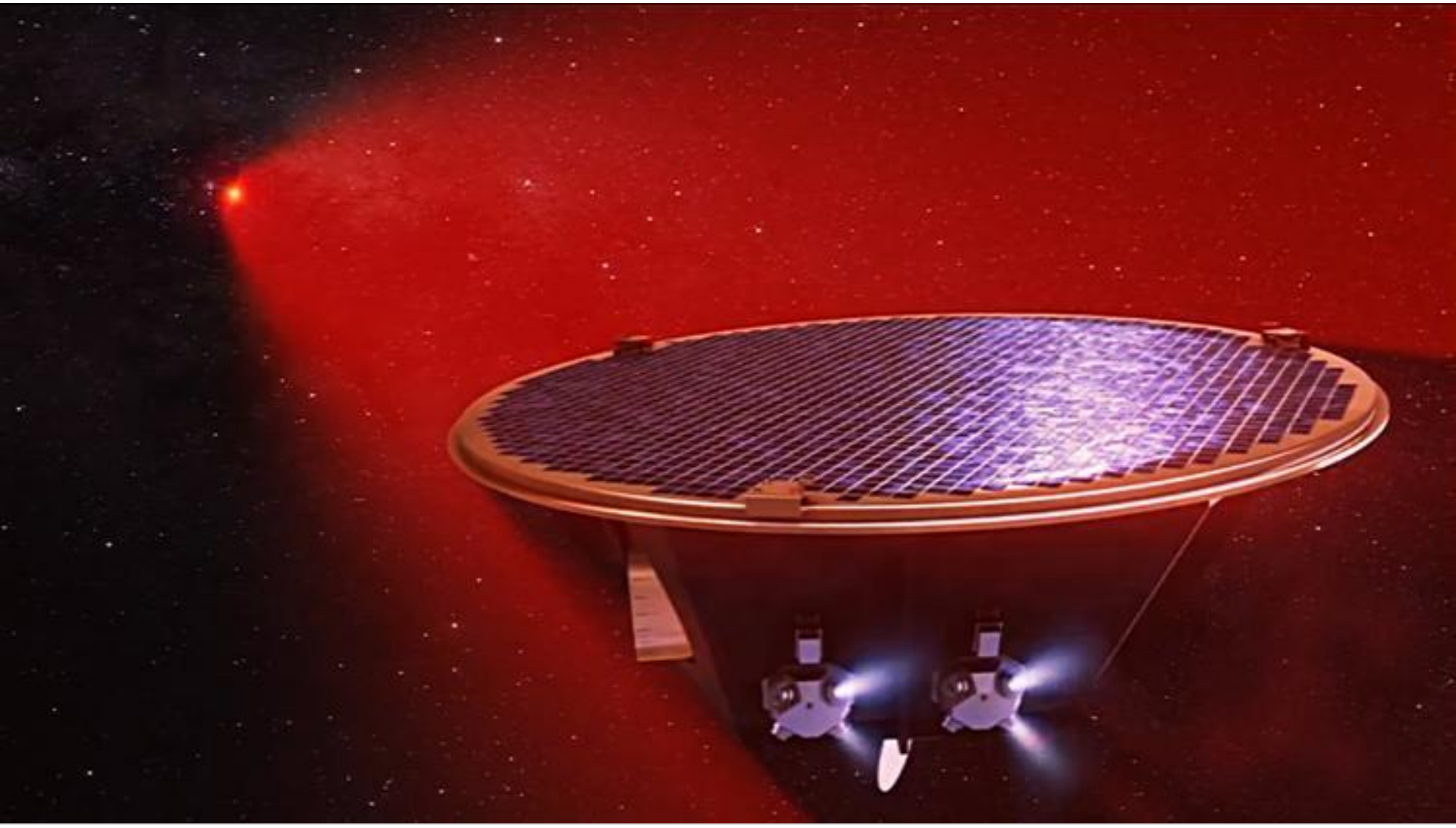



8月26日
12:10


JLG LIFTLUX







LISA will observe a passing gravitational wave directly by measuring the tiny changes in distance between freely falling proof masses inside spacecraft with its high precision measurement system.